Università di Pisa

Artificial Intelligence and Data Engineering

SLAM and Automatic Target Detection Using Neuromorphic Event Camera

Master's Thesis in Artificial Intelligence and Data Engineering

Candidate

Salvatore Arancio Febbo

Supervisors

Prof. Marco Cococcioni

Doc. Niccolò Camarlinghi

Doc. Jose Luis Sanchez Lopez

Anno Accademico 2023-2024

Contents

List of Figures

CHAPTER 1

Introduction

Technological innovation in the field of computer vision has radically transformed how people perceive and interact with their environment. Among these innovations, neuromorphic cameras represent an advanced frontier in the field of artificial vision. These devices are characterized by their ability to detect changes in brightness with extremely high temporal resolution.

This thesis focuses on the use of neuromorphic cameras for Simultaneous Localization and Mapping (SLAM) and automatic target detection. The objectives of this research are:

1. *Find the best open-source framework for SLAM using events from a single neuromorphic camera:* This objective will be achieved through a comparative analysis of various available frameworks, evaluating their capability to handle event data with high precision and reliability.

2. *Identify a second open-source framework for object detection using events:* This objective involves researching and evaluating existing frameworks that can effectively detect objects based on event data from neuromorphic cameras.

3. *Explore the best datasets adaptable for both frameworks:* This objective includes the research and analysis of available datasets that are compatible and useful for training and validating SLAM and object detection frameworks. They will be evaluated based on their representativeness, quality, and variety of data.

4. *Integrate the two frameworks into a ROS system to make it adaptable for the robotic world:* This objective entails developing an integrated system using the Robot Operating System (ROS) to combine SLAM and object detection functionalities, making the sys-

tem applicable to real-world robotic scenarios. The integration will be validated through experiments in real environments.

These objectives have been defined to be SMART (Specific, Measurable, Achievable, Relevant, Time-Bound) and guide the development and validation of the methods proposed in this research.

In *Chapter 2*, the technology of neuromorphic cameras will be detailed, analyzing all their facets and drawing analogies with the human retina, as well as comparisons with RGB cameras.

In *Chapter 3*, an overview of the main event-related datasets will be presented, discussing their limitations and possible solutions.

In *Chapter 4*, the research conducted so far and the state of the art will be examined, with the aim of utilizing this knowledge to achieve the set objectives.

In *Chapter 5*, the proposed method and its evolution will be explored in depth.

The thesis will conclude with *Chapter 6*, which will discuss the results obtained by applying the proposed methods, followed by Conclusions and Future Work as the final chapter.

CHAPTER 2

Introduciton and Details of the Neuromorphic Camera

2.1 Definition and Origins

From its earliest days, humanity has shown an innate inclination towards progress and innovation. A clear example of this tendency is represented by the evolution of photography. In 1839, the *daguerreotype*, created in Paris by the Susse brothers, marked a historical moment with the first photograph. This event induced continuous research and development in image capture technology. This progress has led us to modern *neuromorphic cameras*, first proposed in the 1980s by Carver Mead, but only started to take shape in recent decades. These advanced devices stand out from traditional cameras for their unique features, focusing not on the visual reproduction of the surrounding environment but on capturing the movements and brightness changes of objects.

These tools are rewriting the rules of artificial vision, moving away from traditional frame-based cameras to embrace a technological evolution inspired by the incredible efficiency of the biological visual system (2.1).

These asynchronous devices, finely sensitive to changes in brightness at the pixel level, aim to replicate the process by which the human eye and brain process visual information, ensuring a dynamic and continuous perception of the environment around us. The genesis of this revolutionary technology dates back to the early days of neuromorphic and visual processing research, where the desire to overcome the limitations imposed by traditional cameras inspired the development of sensors capable of interpreting the world not through sequences of static

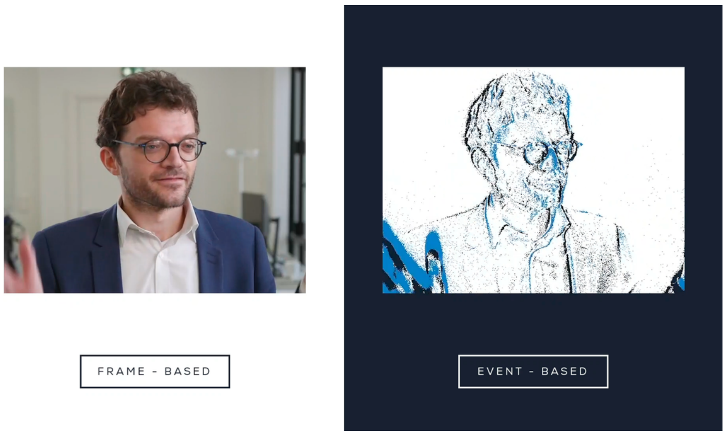


Figure 2.1: *Difference between Vision Sensor and Event Sensor.* [29]

images but as a continuous flow of events. Hence the term *event camera*. This cutting-edge methodology has opened doors to new horizons in visual analysis and interaction, inaugurating an era of discoveries. The pioneering work of *Patrick Lichtsteiner*, *Christoph Posch*, and *Tobi Delbruck*, who first introduced the concept of this device in 2008 [7], laid the foundation for this revolution, marking a decisive moment in the field of visual technology.

2.2 Advantages of the Neuromorphic Camera

The acclaimed article 'Event-Based Vision: A Survey' [32] provides a detailed and comprehensive analysis of the emerging field of event-based vision, highlighting the distinctive features and advantages of event cameras. These advanced technologies are appreciated for their *high temporal resolution*, *low latency*, *wide dynamic range*, and *low power consumption*. As also highlighted in [22], there is also *low bandwidth requirement*, opening new horizons for applications in robotics and artificial vision, in dynamic and challenging environments. We will now illustrate these properties to emphasize the extraordinary capabilities and potential of event cameras, enhancing the reader's understanding of this innovative field of research.

- *Very High Temporal Resolution*: Monitoring brightness changes is rapid, in analog circuits, and event reading is digital, with a clock of 1 MHz, meaning events are detected and timestamped with microsecond resolution. Therefore, event cameras can capture very fast movements without suffering the motion blur typical of frame-based cameras.
- *Low Latency*: Each pixel operates independently and there is no need to wait for a global frame exposure time: as soon as the change is detected, it is transmitted. Therefore, event

cameras have minimal latency: about 10 μs in the laboratory and under a millisecond in the real world.

- *Wide Dynamic Range*: The very wide dynamic range of event cameras (> 120 dB) greatly exceeds the 60 dB of high-quality frame-based cameras, making them capable of capturing information from moonlight to daylight. This is because the pixel photoreceptors operate on a logarithmic scale, offering more detail compared to standard cameras that use a linear scale. Moreover, each pixel operates independently, without waiting for a global shutter. Like biological retinas, DVS pixels can adapt to both very dark and very bright stimuli.

- *Low Power Consumption*: Since event cameras transmit only brightness changes, therefore eliminating redundant data, energy is used only to process changing pixels. At the chip level, most cameras use about 10 mW, and there are prototypes that reach less than 10 μW. Embedded systems with event cameras where the sensor is directly interfaced with a processor have shown a system-level power consumption (i.e., detection plus processing) of 100 mW or less.

- *Low Bandwidth Requirement*: Event sensors generate a stream of events rather than a continuous series of frames, significantly reducing the amount of data produced. This aspect is particularly advantageous for applications in sensor networks where bandwidth is limited.

Humanity has always had the ambition to create artificial replicas of itself, a drive that guides technological advancement towards the development of machines with perceptions increasingly close to human ones. This camera, with its advanced visual characteristics, has the potential to come closer than ever to achieving this goal.

2.3 Applications of the Neuromorphic Camera

The neuromorphic camera finds applications in various fields, including:

- *Artificial Vision and Robotics*: Event cameras are applied in artificial vision and robotics, where their ability to capture rapid movements and lighting changes can be used to improve robots' orientation, navigation, and object manipulation.[32]

- *High-Speed Motion Tracking*: The unique capabilities of event cameras make them ideal for high-speed motion tracking, such as monitoring fast-moving objects or sports events, where they can provide precise data without the motion blur typical of traditional cameras.[31] [24]

- *Depth Perception and 3D Reconstruction*: The use of event cameras for depth perception and 3D reconstruction leverages their high acquisition frequency and edge sensitivity of objects to generate detailed maps of environments using stereo cameras.[19]

- *Enhanced Vision in Low Light Conditions*: Thanks to their wide dynamic range and sensitivity to brightness changes, event cameras are particularly suitable for vision applications in low light conditions, such as night driving or surveillance in poorly lit environments.[31]

Research and development in this field are advancing rapidly, opening new horizons and application possibilities. The cited publications provide a broad overview of the potential of event cameras and the ongoing efforts of the scientific community to explore and exploit these innovative technologies. However, despite significant progress, challenges remain, particularly regarding the integration of these technologies into more complex systems and their optimization for specific applications.

2.4 Limitations of Event-Based Cameras

After examining the advantages, it is essential to also consider the other aspect of the issue, the disadvantages, which can prove to be the most critical elements. Understanding the disadvantages is crucial because it offers us the opportunity to significantly improve the results achieved thanks to the advantages. This awareness allows us to refine and optimize our strategies, thus enhancing the overall effectiveness of our actions.

2.4.1 Intrinsic Technical Limitations

The document "Event-Based Vision: A Survey" [31] not only highlights the advantages but also shows us the technical limitations of event-based cameras.

- *Managing Different Outputs:* The output produced by event-based cameras is remarkably different from that generated by traditional cameras: the first ones record asynchronous and spatially distributed events, while the others capture synchronous images characterized by a dense distribution. Therefore, visual algorithms operating on frames and designed to work with image sequences are not suitable for processing event camera data.

- *Different Photometric Perception:* Unlike traditional cameras, which provide grayscale data, each event detected by event cameras is associated with a binary information of brightness variation (increase or decrease). These variations are influenced not only by the environment brightness but also by the relative movement between the observed scene and the camera.

- *Noise and Dynamic Effects:* All optical sensors are subject to noise (photon and transistor circuit noise) and present imperfections. This is particularly evident in event cameras, which face the complexity of temporal contrast quantization, a process that has not yet been fully described and understood.

2.4.2 Practical Limitations

When exploring the practical restrictions and complexities related to the processing and analysis of data obtained from these cameras, there are several challenges that require detailed attention:

- *Data Accessibility:* The difficulty in finding particularly useful data for a specific problem online is one of the main challenges. This aspect will be further explored in section 3.2, where we will analyze possible solutions to this problem.

- *Data Variability:* The uniqueness of each data set, due to the specific configurations of each camera, both at the hardware and software level, introduces further complexity. This includes:

  - Hardware configurations, which can vary significantly between devices.
  - Software settings, which influence the data acquisition and analysis process.

Despite these challenges, the Metavision framework [44], developed by Prophesee, is making significant progress towards standardizing data in event cameras. This development promises to simplify data analysis and improve accessibility for researchers in this field.

For the moment, we definitely face several difficulties. Moving forward, we will delve into specific techniques to address these challenges. We will also explore strategies to leverage the advantages and mitigate the disadvantages, aiming to maximize the effectiveness of this system.

2.5 Hardware Analogies between the Human Retina and DVS Pixels

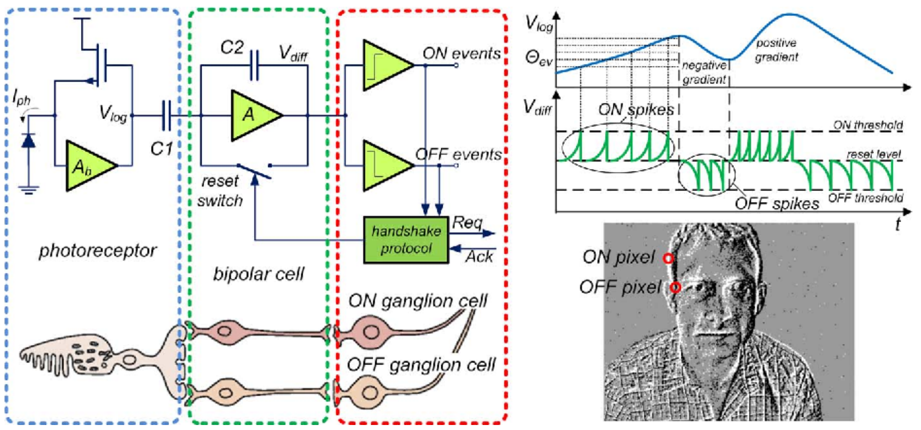


Figure 2.2: *Retinomorphic Event-Based Vision Sensors: Bioinspired Cameras With Spiking Output* by C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco, and T. Delbrück, published in Proceedings of the IEEE in 2014.[9]

The technology of Dynamic Vision Sensors *(DVS)* bridges the gap between the complexity of human vision and cutting-edge engineering, mimicking the functionality of the three main layers of the retina: *photoreceptors*, *bipolar cells*, and *ganglion cells* 2.2. These sensors capture light changes through a pixel circuit that reacts with spike events to changes in light, reflecting the process of converting and transmitting visual signals from the natural world to the human brain. In detail, each DVS pixel tracks light changes similarly to human photoreceptors, generating events of different polarity for positive or negative changes, similar to the role of bipolar cells. These events are then processed and transmitted in a manner reminiscent of ganglion cells, carrying visual information to the brain or the next processing stage. DVS technology excels in dynamically encoding scenes, transforming intensity variations into a temporal stream of events, providing an efficient representation of movements and light transitions in the observed scene. This analogy between human visual mechanisms and DVS circuits highlights the importance of biological inspiration in technological advancements. Implementing such sensors promises more responsive and efficient artificial vision systems, capable of quickly adapting to environmental changes, with applications ranging from pattern recognition to advanced robotics, demonstrating the potential of this synergy between biology and technology.

2.6 Comparison between Neuromorphic Camera and RGB Camera

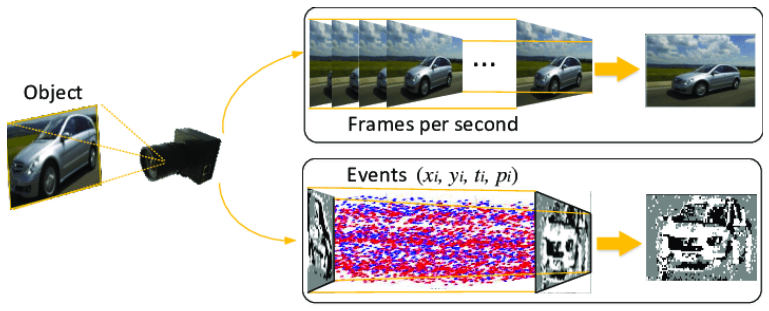


Figure 2.3: *An End-to-End Broad Learning System for Event-Based Object Classification*

Before delving into the actual use of the neuromorphic camera, it is appropriate to compare it with the standard cameras we use daily through our smartphones, commonly known as *RGB* cameras. But why "RGB"? An *RGB camera* is called so because it captures images using a sensor that records light in three primary color channels: R (*Red*), G (*Green*), and B (*Blue*). These three primary colors, when combined in different proportions, can create a wide range of colors.

When light hits the camera sensor, the corresponding photodiodes in the three color channels record the light intensity in each of the three primary colors. These intensities are then combined to form a color image. When recording videos, an RGB camera, simplifying the concept, captures a series of images very close to each other every second, called *frames*. These frames are sequenced one after the other to create the illusion of movement.

As shown in figure (2.3), while a conventional camera typically records at a frequency of 30/60 frames per second, a single pixel of an event-based camera can reach a frequency of over 60,000 events per second [7]. It is important to note that frames capture whole images at regular intervals, regardless of changes in the scene, while events record only brightness changes at specific points in the scene and asynchronously, thus offering a more efficient and detailed representation of temporal variations.

To summarize the main differences between neuromorphic cameras and traditional RGB cameras, the following comparison table is presented:

Feature	*Neuromorphic Camera*	*RGB Camera*
Acquisition Frequency	Over 60,000 events per second per pixel	30/60 frames per second
Acquisition Method	Asynchronous, records only brightness changes	Synchronous, captures whole images at regular intervals
Energy Efficiency	About 10 mW (prototypes < 10 µW), complete system < 100 mW	Hundreds of milliwatts to several watts
Resolution	Varies, often lower than RGB cameras	High, millions of pixels
Light Sensitivity	High, sensitive to brightness variations	Variable, depends on sensor quality
Types of Data Recorded	Brightness changes	Color images

Table 2.1: Comparison between Neuromorphic Camera and RGB Camera

2.7 Devices and Companies that Produce Them

Below are the main companies that produce these sensors and the characteristics of their flagship devices, aiming to approach the practical aspect.

2.7.1 Main Companies

- *iniVation AG*
  - iniVation AG specializes in neuromorphic technologies with a focus on event-based vision. It offers a range of Dynamic Vision Sensors (DVS) and Hybrid Vision Sensors (HVS), promoting the integration of these technologies in industrial and commercial sectors.
- *Samsung Electronics*
  - Samsung has developed several Dynamic Vision Sensors (DVS) with various resolutions and applications. Samsung is known for its advancements in miniaturization

and performance optimization of sensors for use in low-latency environments.

- *Prophesee*
  - Prophesee is a leader in event-based vision technology, offering advanced sensors like Metavision® IMX636 and IMX646. Their technology is used in a variety of applications, from industrial automation to advanced robotics.
- *Sony*
  - Sony has collaborated with other companies like Prophesee to develop high-performance event-based vision sensors. Their technology stands out for its high resolution and ability to operate in low-light conditions.
- *CelePixel*
  - CelePixel is an emerging company specializing in high-resolution event-based vision sensors, such as the CeleX-V, which is the first 1-megapixel event sensor.

2.7.2 Devices and Features

Before delving into the details of specific devices, it is useful to understand some of the main features that distinguish these event-based vision sensors:

- *Resolution*: Indicates the total number of pixels that make up the image, directly influencing the clarity and visible details.
- *Optical Format*: Refers to the physical size of the image sensor, influencing the amount of light captured and the image quality in low-light conditions.
- *Pixel Latency*: Indicates the time it takes for a pixel to change state in response to a light variation. Lower values indicate a better ability to capture fast movements without blur.
- *Dynamic Range*: Measures the sensor's ability to capture details in very bright and very dark areas of the scene.
- *Pixel Size*: Determines the amount of light that can be absorbed, improving image quality in low-light conditions.

DVS (Dynamic Vision Sensor)

- *Produced by*: iniVation AG
- *Features*:
  - Resolution: 128x128
  - Dynamic Range: 120dB
  - Latency: 15s
  - Asynchronous temporal vision technology, capturing light contrast variations.

DAVIS (Dynamic and Active Pixel Vision Sensor)

- *Produced by*: iniVation AG
- *Features*:
  - Resolution: 240x180
  - Dynamic Range: 130dB
  - Latency: 3s
  - Combines dynamic vision with active pixels to offer both brightness variation events and static images.

Metavision® IMX636 and IMX646

- *Produced by*: Prophesee in collaboration with Sony
- *Features*:
  - Resolution: 1280x720
  - Optical Format:
    1/2.5"
  - Pixel Latency: <100s
  - Dynamic Range: >120dB
  - Ideal for industrial and mobile applications, with high speed and ability to operate in low-light conditions.

Samsung DVS

- *Produced by*: Samsung Electronics
- *Features*:
  - Resolution: Up to 1280x960
  - Dynamic Range: >120dB
  - Pixel Latency: <1s
  - Sensors designed to minimize motion artifacts, used in home monitoring and robotics applications.

CeleX-V

- *Produced by*: CelePixel
- *Features*:
  - Resolution: 1 Megapixel
  - Dynamic Range: >120dB
  - Sensitivity to light changes for applications in augmented reality (AR) and virtual reality (VR).

CHAPTER 3

Overview of Available Datasets

3.1 Description of Main Datasets

This chapter provides a detailed overview of the main datasets available for research and development with event cameras, including the DVS128 Gesture Dataset, the N-CARS Dataset, the N-MNIST and N-Caltech101 Datasets, the DDD17 Dataset, the EV-IMO Dataset, as well as the more recent 1 Mpx Event Camera Dataset and the GEN1 Automotive Detection Dataset. For a summary of these datasets, please refer directly to Table 3.1.

3.1.1 DVS128 Gesture Dataset

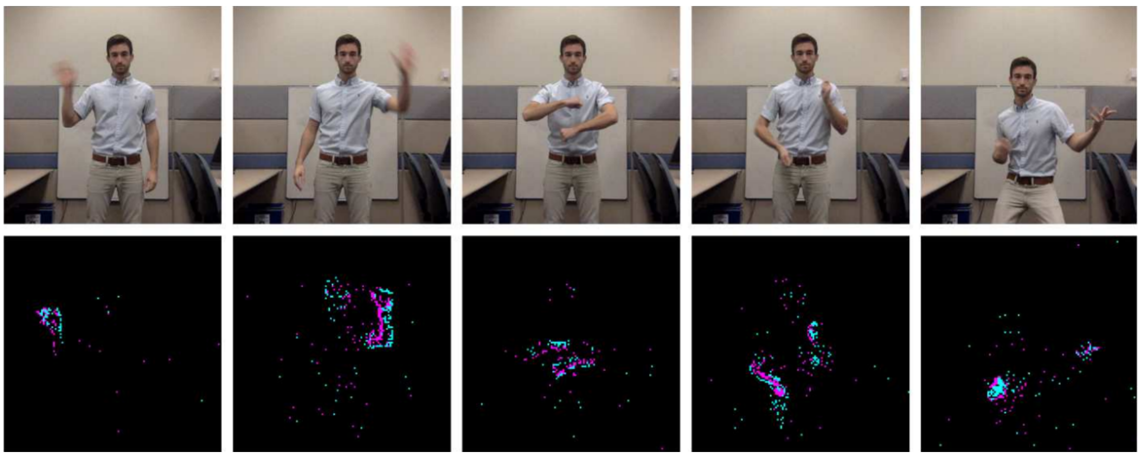


Figure 3.1: DVS128 gesture [17]

The DvsGesture dataset [11] consists of 1,342 instances of 11 manual and arm gestures (Fig. 7), organized into 122 trials collected from 29 subjects under 3 different lighting conditions.

In each trial, a subject stood against a stationary background and performed all 11 gestures sequentially under the same lighting condition. The gestures include hand waving (both arms), large straight arm rotations (both arms, clockwise and counterclockwise), forearm rolling (forward and backward), air guitar, air drums, and a "Other" gesture invented by the subject. The 3 lighting conditions are combinations of natural light, fluorescent light, and LED light, selected to control the effect of shadows and fluorescent light interference on the DVS128. Each gesture lasts approximately 6 seconds.

3.1.2 N-CARS Dataset

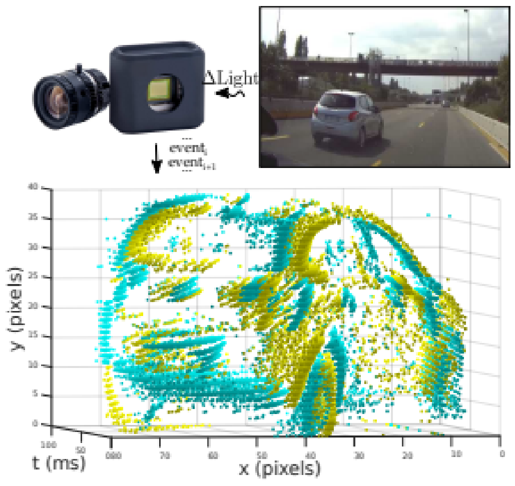


Figure 3.2: N-CARS [20]

N-CARS [20] is a large public dataset used for car classification. This real-world dataset consists of 12,336 car samples and 11,693 background samples. During data collection, the event camera was mounted behind the windshield of a moving car, recording exactly 100 milliseconds of data per sample, with a variable number of events ranging from 500 to 59,249 per sample. The abundance of samples and the diversity within each class make this dataset particularly suitable for machine learning. However, its containing only two classes reduces the complexity of the challenge it presents.

3.1.3 N-MNIST and N-Caltech101 Datasets

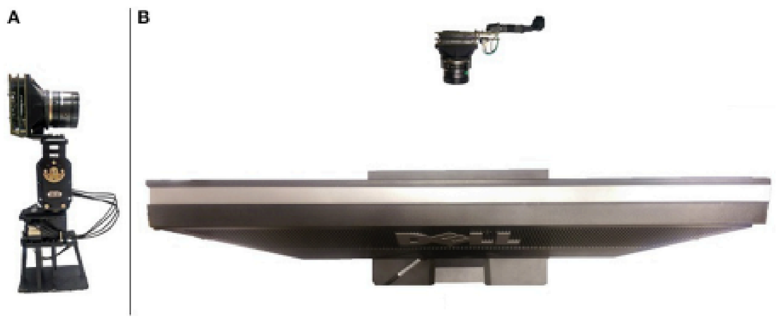


Figure 3.3: *(A)* A picture of the ATIS mounted on the pan tilt unit used in the conversion system. *(B)* The ATIS placed viewing the LCD monitor.[12]

MNIST and Caltech101 are two fundamental datasets for machine learning, specializing in handwritten digit recognition and object classification in images, respectively. While MNIST focuses on black and white images of digits from 0 to 9, Caltech101 features a variety of object categories with color images. Both datasets have played a crucial role in the development of artificial vision algorithms.

To simulate visual perception through event cameras, N-MNIST and N-Caltech101 respectively transform the MNIST and Caltech101 datasets by simulating saccadic eye movements on static images (3.4). This method captures visual dynamics that an ATIS sensor would record (3.3) in real scenarios, generating event patterns for each image. N-Caltech101 extends this approach to the various object categories in Caltech101, offering a new dimension for research in the field of neuromorphic computational vision.

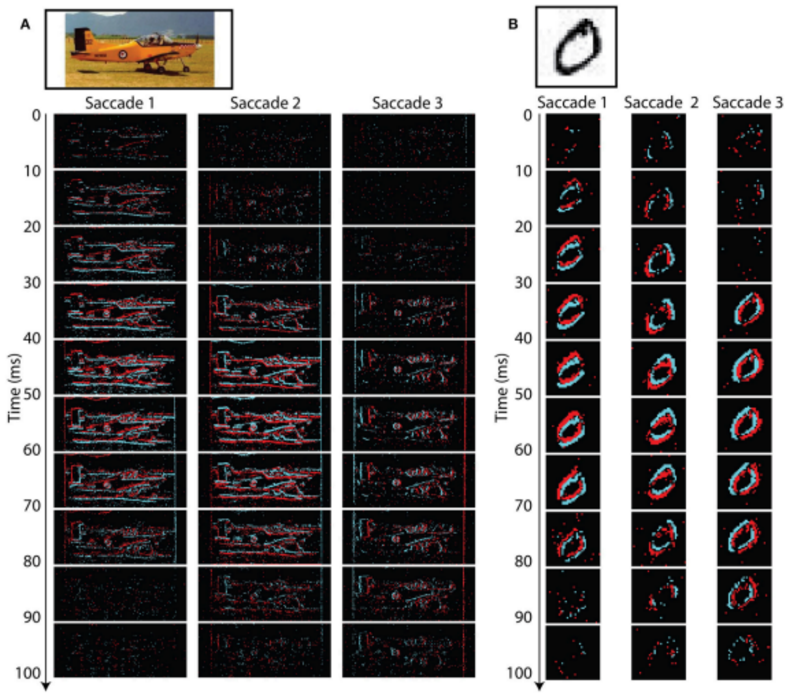


Figure 3.4: *Typical recordings for Caltech101 (A) and MNIST (B).* The original images are shown at the top, and the recorded events obtained during each saccade are shown below. Each event image shows 10 ms worth of data. Black regions indicate no events, red indicates ON events, and blue indicates OFF events. The third saccade is the shortest in distance and therefore generates the fewest events. Fewer events are recorded near the start and end of each saccade when the ATIS is moving slowest.[12]

3.1.4 DDD17: End-to-End DAVIS Driving Dataset

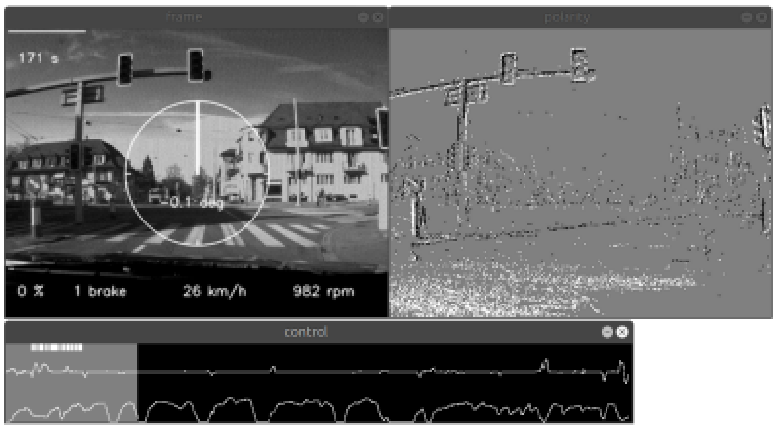


Figure 3.5: Example scenario visualized by the recording file viewer. The top panels show the DAVIS frames (left; overlaid with some driving data) and events (right), the bottom panel shows a progress bar as well as visualizations of different vehicle data (headlamp status at the top, steering angle in the middle, speed at the bottom). [18].

DDD17 [18] stands as a significant advancement in autonomous driving research, being the first open dataset with annotated driving recordings from a DAVIS camera. This dataset encompasses over 12 hours of footage, capturing driving under various lighting and weather conditions, in both urban and highway settings. It includes essential data such as GPS location, vehicle speed, and driver actions like steering, acceleration, and braking, offering a valuable resource for the development of advanced driver-assistance technologies.

3.1.5 EV-IMO Dataset

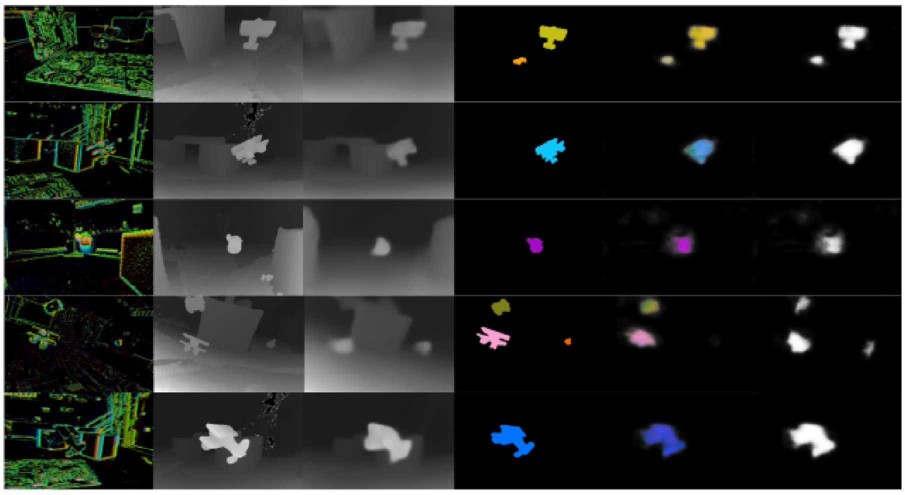


Figure 3.6: The table entries from left to right: DVS input, ground truth for depth, network output for depth, ground truth pixel-wise pose, predicted pixel-wise pose, predicted motion mask. Examples were collected from EV-IMO dataset. Best viewed in color. [23].

The EV-IMO dataset consists of 32 minutes of indoor video recordings, characterized by the presence of 1 to 3 fast-moving objects captured simultaneously by the camera. The objects and the camera are accurately tracked using the VICON motion capture system[1]. Utilizing 3D scans of the environments and objects, the dataset offers accurate depth maps and pixel-level object motion masks, reliable even in poor lighting conditions or during fast movements.

[1]VICON motion capture system are cutting-edge technologies used to capture precise movements of objects or people in three-dimensional space.

3.1.6 1 Mpx Event Camera Dataset

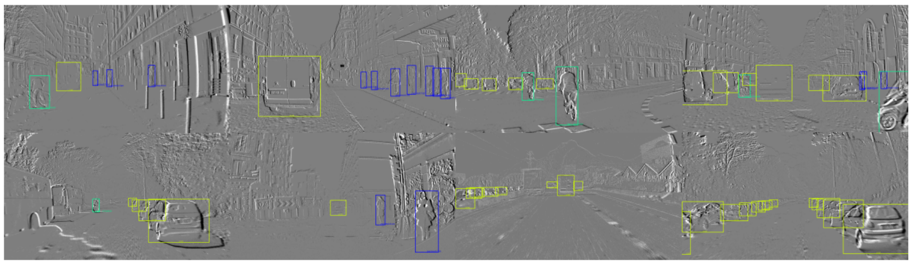


Figure 3.7: The event-camera detection method is able to accurately detect a wide range of objects, regardless of their appearance, the context in which they are located, and the speed at which they move. Specifically, the detected cars, pedestrians, and motorcycles are highlighted with yellow, blue, and cyan boxes respectively. [28].

1 Mpx is the first large-scale high-resolution dataset for object detection. This dataset contains over 14 hours of recordings from a 1-megapixel event camera, used in automotive scenarios. Additionally, the dataset includes 25 million bounding boxes, which are delineated regions surrounding objects of interest, such as *cars*, *pedestrians*, and *motorcycles*. These bounding boxes have been labeled at high frequency, meaning that each object has been annotated with great precision and detail.

3.1.7 GEN1 Automotive Detection Dataset

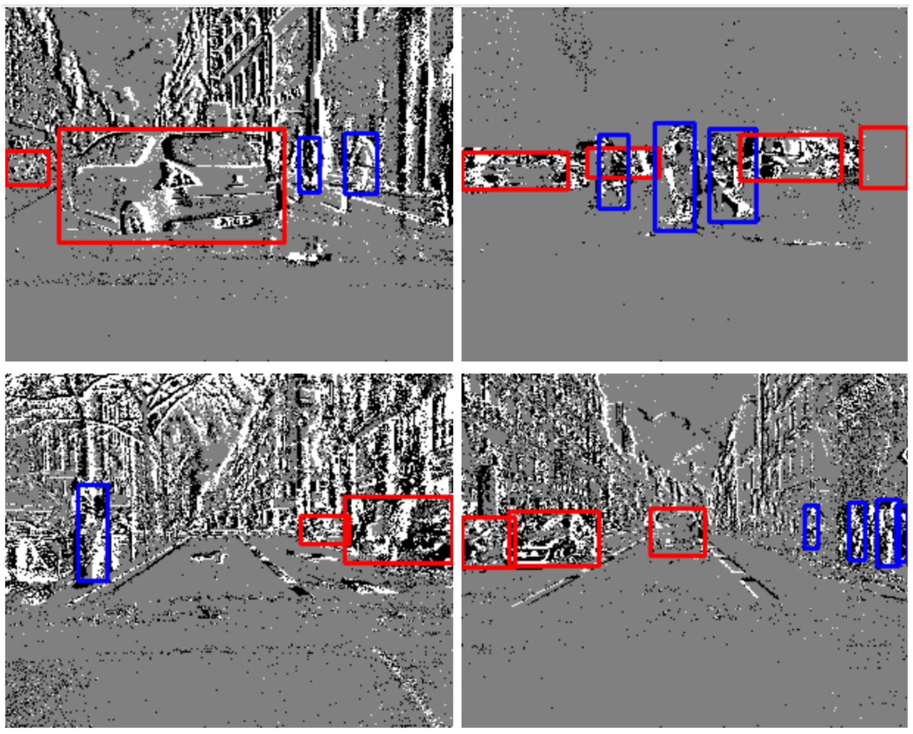


Figure 3.8: GEN1: Blue bounding boxes correspond to pedestrian labels, red bounding boxes to cars. [25].

Gen1 Automotive Detection Dataset consists of over 39 hours of automotive recordings acquired with a 304x240 GEN1 sensor. It includes open roads and highly diverse driving scenarios, ranging from *urban*, *highway*, *suburban*, and *rural* scenes, as well as various weather and lighting conditions. Additionally, manual annotations of *bounding boxes* for *cars* and *pedestrians* present in the recordings are provided at a frequency ranging from 1 to 4Hz, totaling over 255,000 labels.

Dataset	**Description**	**Key Features**	**Target**
DVS128 Gesture Dataset	Consists of manual and arm gestures performed under various lighting conditions.	-1,342 instances of 11 gestures -122 trials from 29 subjects -3 lighting conditions	Gesture Recognition
N-CARS Dataset	Used for car classification with samples collected from a moving car.	-12,336 car samples -11,693 background samples -100 milliseconds of data per sample	Car/Background Classification
N-MNIST and N-Caltech101	Simulates saccadic eye movements on static images to transform MNIST and Caltech101 datasets for neuromorphic vision.	-Simulated visual dynamics -Focus on handwritten digit recognition (N-MNIST) and object classification (N-Caltech101)	Digit Recognition/Object Classification
DDD17 Dataset	Annotated driving recordings from a DAVIS camera covering various conditions.	-Over 12 hours of footage -GPS data, vehicle speed, steering, acceleration, and braking information	Autonomous Driving Assistance
EV-IMO Dataset	Indoor video recordings with fast-moving objects and accurate tracking.	-32 minutes of footage -Accurate depth maps and object motion masks at the pixel level	Motion Segmentation
1 Mpx Event Camera Dataset	First large-scale high-resolution dataset for object detection in automotive scenarios.	-Over 14 hours of recordings -25 million bounding boxes for cars, pedestrians, and motorcycles	Object Detection in Automotive Scenarios
GEN1 Automotive Detection Dataset	Over 39 hours of automotive recordings with diverse driving scenarios and weather conditions, annotated for cars and pedestrians.	-Over 39 hours of footage -More than 255,000 labels for cars and pedestrians	Object Detection of Cars and Pedestrians in Outdoor Environments

Table 3.1: Summary of the main datasets for event cameras and neuromorphic vision.

3.2 Issues and Limitations of Existing Datasets

Even though there are more and more datasets available for *event cameras*, there are several issues and limitations that can affect how effective and applicable machine learning and *computer vision* models are when developed on this data. These problems range from the variety and representativeness of the data to their size and quality, directly influencing the *generalizability* and efficiency of artificial intelligence systems based on these datasets.

3.2.1 Data Representativeness and Variability

One of the most significant obstacles in the field of computer vision concerns the representativeness of datasets, particularly evident in the case of *event cameras*. Data collection, often conducted in highly controlled environments or confined spaces, fails to capture the wide spectrum of variability and complexity that characterizes the real world. This limited scenic panorama, while facilitating the collection of a large volume of data, induces a certain homogeneity that risks distorting the reality it aims to model.

In practice, the acquisition methodology tends to favor quantity over diversity, adopting strategies that *force* event cameras to capture data, often repeating the same types of scenarios. Although this practice is useful for maximizing collection efficiency, it introduces a significant *bias* in the dataset: a narrow representation of the world that inevitably limits the generalization capability of machine learning models developed on such a basis.

This can lead to a reduced ability of the models to generalize to new contexts or to handle situations not previously encountered during the training phase.

3.2.2 Data Size and Quality

The *size of datasets* represents another significant challenge. Many event datasets tend to be relatively small compared with the standard data sets used for training deep learning models, thus limiting the models' ability to learn complex features and subtle variations in the data. Beyond size, the *quality of data* plays a crucial role, varying considerably based on the specific hardware configurations used for capture and the prevailing environmental conditions during the recording process. These variables can significantly influence the clarity and accuracy of the captured events, with tangible reflections on the actual usefulness of the data for research purposes and practical applications. For example, a recording made in conditions of poor lighting or with one event camera rather than another with different characteristics can result in lower quality data, making it difficult for models to learn relevant features or perform accurate classification tasks.

3.2.3 Data Imbalances

Many datasets show significant imbalances in label classes, where some categories are much more represented than others. This disparity can introduce bias in machine learning models, leading to better performance on the overrepresented classes at the expense of the less frequent ones. This phenomenon, along with the issues discussed earlier, further compromises the AI's ability to generalize and function effectively in diverse real-world scenarios.

3.2.4 Lack of Standardization

The question of *standardization* emerges as a central theme in the context of the challenges associated with datasets for event cameras. Problems such as the variety in data representativeness, fluctuations in their quality, and differences in acquisition modes are all symptoms of a fundamental need: the implementation of uniform standards. In the absence of a consensus on best practices and technical specifications, companies continue to produce hardware according to disparate guidelines. This, in the context of a relatively young sector like that of event cameras, further complicates research and development.

The field of event cameras is still in a phase of exploration and innovation. Much remains to be discovered and optimized, from sensor sensitivity to data processing modes. The adoption of a shared standard could not only facilitate compatibility between different systems but also direct the community towards common goals of quality and efficiency improvement.

The proposal of a regulatory entity, or a consortium of industries and academics, that guides the development and adoption of standardized protocols, could significantly accelerate technological development in the sector. Such an organism would have the task of defining clear guidelines for hardware production, data formats, and annotation procedures, ensuring that progress in the field of event cameras proceeds in a more harmonious and coordinated manner.

3.2.5 Awaiting Possible Standardization and Subsequently a Homogeneous and Genuine Data Increase

While we wait for a possible standardization and subsequently a genuine and uniform increase in data, an important necessity emerges: to adopt a flexible approach capable of responding, even if only partially, to the manifold needs of the field. This involves outlining a *general method* that, avoiding getting lost in overly specific details, can propose effective solutions to the various problems encountered. Ideally, we would identify a strategy that, with a reasonable effort, is capable of producing concrete and valuable results for the research community.

This approach, as one might intuitively understand, focuses on the manipulation and processing of data. But how exactly? This question will be the subject of a more detailed discussion in the next chapter, where we will explore the techniques and strategies to optimize the use of the data at our disposal, turning the current challenges into opportunities for scientific and technological progress.

3.3 Methods for Dataset Enrichment

To address challenges related to the quality, variety, and representativeness of event datasets, research has developed several methods to enrich and enhance these datasets. These approaches not only aim to increase the effectiveness of machine learning and computer vision models but also strive to make the datasets more representative of real-world usage scenarios.

3.3.1 Data Augmentation for Event-based Datasets

EventDrop

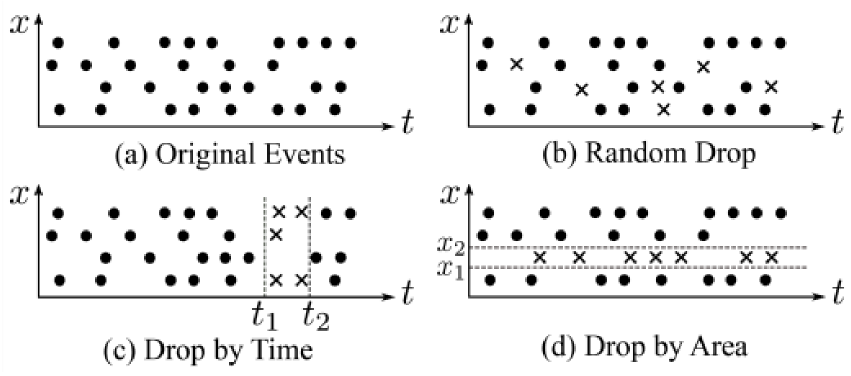


Figure 3.9: : Strategies used by EventDrop, where t indicates time dimension, x denotes the pixel coordinate (only one dimension is shown here for clarity). Dots represent original events, and × denotes the events to be dropped. Dashed lines represent threshold borders. (a) Original events that are triggered asynchronously. (b) Random drop strategy. (c) Drop by time strategy. (d) Drop by area strategy[30].

Data augmentation plays a crucial role in improving the generalization of deep learning models. Recently, Gu et al. introduced an innovative method specifically designed for augmenting event-based datasets, called *EventDrop*. As shown in (3.9) this technique employs three distinct augmentation strategies:

1. *Random Drop*: randomly removes events from the dataset.
2. *Drop by Time*: deletes all events within a specified time period
3. *Drop by Area*: erases events from specific areas of the event matrix

These experiments were conducted on two event-based datasets, N-Caltech101 and N-Cars, pre-

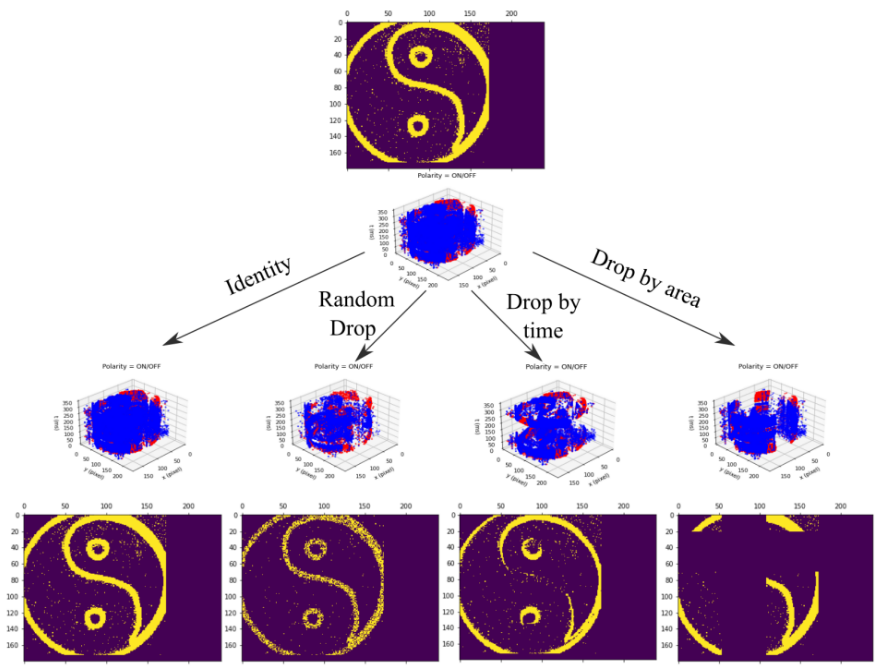


Figure 3.10: An example of augmented events with EventDrop. For better visualization, the event frame representation is used to visualize the outcome of augmented events[30].

viously discussed in the preceding sections, and it was demonstrated that EventDrop contributes to a significant improvement in generalization capability across various deep networks. The paper [30] demonstrates how the adoption of this technique has improved the accuracy in object classification, resulting in an increase of up to 4% compared to previous performance.

NDA

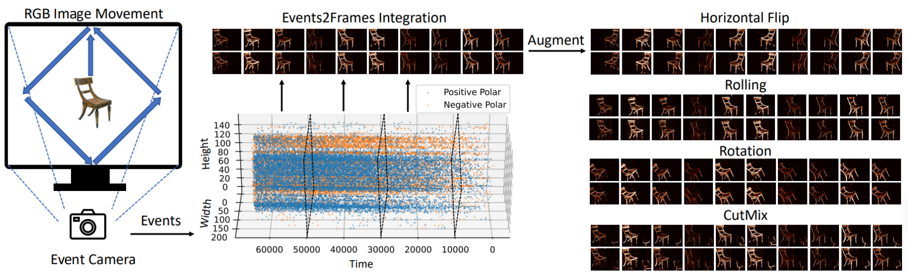


Figure 3.11: Left: Event data collection of a chair image from N-Caltech101 [47], these events are too sparse to process, thus we integrate them into frames of sparse tensors. Right: Our neuromorphic data augmentation on events data[34].

At the same time as EventDrop, an augmentation strategy named NDA (Neural Data Augmentation) [30] was developed, specifically designed for event-based datasets, which incorporates a range of geometric augmentation techniques. These techniques, illustrated in the figure (3.11), include:

1. *Horizontal Flipping*: A technique widely used in computer vision tasks, which reverses the order of the data's horizontal dimensions, keeping the polarity and temporal dimension of the events intact.
2. *Rolling*: Involves randomly shifting the geometric position of the DVS image, both left and right in the horizontal dimension, and up and down in the vertical dimension. The shifts can be circular or not.
3. *Rotation*: The DVS data are randomly rotated in a clockwise or counterclockwise direction.
4. *Cutout*: Originally proposed to combat overfitting, this method randomly erases a small area of the image, similarly to the effect of dropout. A square of random size and position is generated, and all pixels within this square are masked.
5. *Shear*: Shear mapping originated from plane geometry. It is a linear map that displaces each point in a fixed direction. Mathematically, a horizontal shear (or ShearX) is a function that takes point (x, y) to point (x + my, y). All pixels above the x-axis are displaced to the right if $m > 0$, and to the left if $m < 0$. Note that vertical shear (ShearY) is not used, following prior art.
6. *CutMix*: An effective linear interpolation of two data samples and their corresponding labels, proposed to combine information from two different inputs.

Applying these techniques to the N-Caltech 101 dataset, it was possible to significantly increase accuracy, with an impressive absolute improvement of 15.8%. On the N-MNIST dataset, NDA increased the models' accuracy by 0.12%. These results highlight the effectiveness of geometric augmentation strategies in enhancing the performance of deep learning models on event-based datasets, offering a significant contribution to the field of computer vision and machine learning.

The idea of combining the EventDrop strategy with that of NDA to enhance event-based datasets is truly intriguing and could offer new perspectives for advancing deep learning models. This hybrid approach could not only amplify the diversity of data but also simulate more complex and variable conditions, thereby significantly enhancing the robustness and generalization of models.

3.3.2 Synthetic Events through Video Conversion

In the previous section, we discussed the concept of Augmentation, showing how this technique can expand our datasets. However, we face a significant challenge: the limited variety of datasets available. Despite the ability to augment our data, Augmentation primarily focuses on modifying the position or appearance of objects without altering the context they are in. For example, no matter how many changes are made to a video of a dog running on a beach, the main subject and its context remain unchanged. This limitation becomes especially apparent when

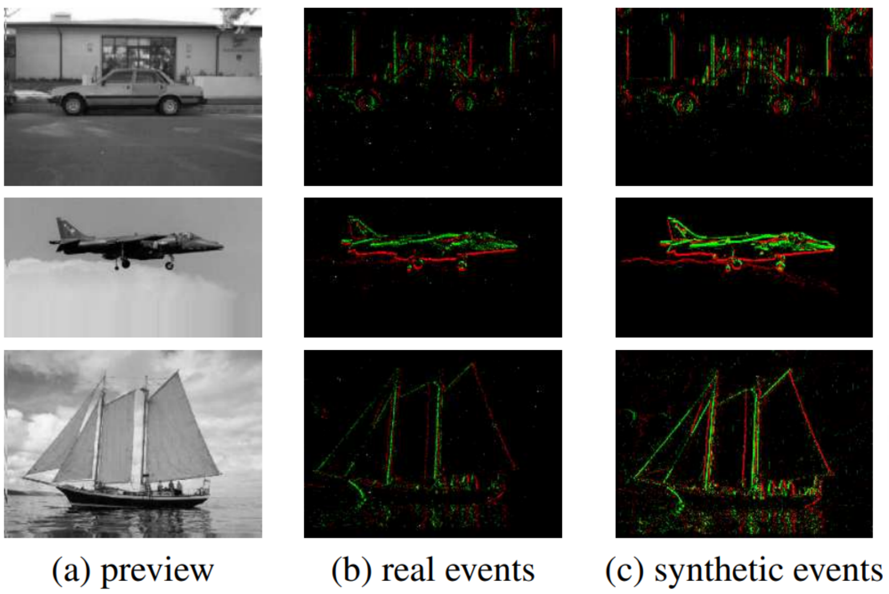


Figure 3.12: A side-by-side comparison of a samples from Caltech101 (a), N-Caltech101 (b) and our synthetic examples from (c) simN-Caltech101. While the real events were recorded by moving an event camera in front of a projector, the synthetic events were generated using ESIM by moving a virtual camera in front of a 2D projection of the sample in (a) [26].

considering the use of standard datasets of videos and photographs in applications that require a variety of contexts. Unlike what happens in traditional image datasets, where the variety of contexts is often sufficient, in the world of event cameras the diversity of context is crucial but lacking.

An innovative approach to overcome this obstacle has been introduced by Gehrig et al. [26], proposing a methodology to transform existing video datasets into synthetic event data. This method leverages the unique ability of event cameras to record changes in brightness with high temporal precision, generating synthetic data that simulates the characteristics of real events as we can see in the photo (3.12). The importance of this technique lies in its ability to use the wide repertoire of available videos, turning them into valuable resources for training and testing models in the context of artificial vision related to event cameras.

The authors demonstrate that models trained on these synthetic data can achieve performance comparable to those trained on real event datasets. This represents a significant step towards solving the problem of the lack of contextual variety in event datasets, paving the way for a wide range of innovative applications in the field of artificial vision. The possibility of generating synthetic event data from standard videos not only facilitates the development of new algorithms for event cameras but also expands the research and application prospects in this emerging field.

3.3.3 Crowdsourcing and Community Collaborations

We have explored how to expand our dataset by drawing from existing events and integrating data from standard cameras to identify new events, demonstrating that it is theoretically and artificially feasible to generate the desired but not yet existing data. However, it is crucial to highlight the importance of data collected directly from reality, which, devoid of the manipulations described above, represent a more authentic and preferable source.

To inform the reader, there is a superior alternative that leverages crowdsourcing and community collaborations. This approach, although requiring standardization and time, invites a wide community of researchers and enthusiasts to share their data on events, allowing us to obtain a dataset that is not only more varied but also more representative of different scenarios and contexts. The implementation of clear standards for data collection and annotation is essential to ensure the consistency and usability of contributions.

Adopting this strategy brings significant advantages: in addition to increasing the quantity and quality of available data, it promotes a spirit of cooperation and knowledge sharing within the community. This not only enriches the dataset at our disposal but can also accelerate progress in research and development of event-based vision applications, highlighting the importance of community collaboration in the field.

CHAPTER 4

State of the Art of Object Detection and SLAM with Solutions to Dataset Limitation

In this chapter, the models that best meet our requirements and have been selected based on their state-of-the-art performance, both in object detection and SLAM, will be examined.

4.1 Object Detection Models used in the Project

Right after studying the events, the study of public models was investigated to understand the pros and cons of each and in which situations they might be suitable. In this section, the two event object detection models that currently represent the state of the art are discussed.

The metrics used for the selection of the best model to use were mAP (mean Average Precision), inference time, and model training time.

The mean Average Precision (mAP) is defined as follows:

$$\text{mAP} = \frac{1}{|Q|} \sum_{q \in Q} AP(q) \tag{4.1.1}$$

Where Q is the set of all queries and $AP(q)$ is the average precision for query q.

The Average Precision (AP) for a single query is the average of the precision values at the ranks where relevant items are found. It is defined as:

$$\text{AP}(q) = \sum_{k=1}^{N} (R_k - R_{k-1}) P(k) \tag{4.1.2}$$

Where:

- N is the total number of thresholds considered.
- R_k is the recall at cutoff k.
- $P(k)$ is the precision at cutoff k.

Precision at cutoff k is defined as:

$$P(k) = \frac{T_p(k)}{T_p(k) + F_p(k)} \tag{4.1.3}$$

Where:

- $T_p(k)$ is the number of true positives at cutoff k.
- $F_p(k)$ is the number of false positives at cutoff k.

Recall at cutoff k is defined as:

$$R_k = \frac{T_p(k)}{T_p(k) + F_n(k)} \tag{4.1.4}$$

Where:

- $F_n(k)$ is the number of false negatives at cutoff k.

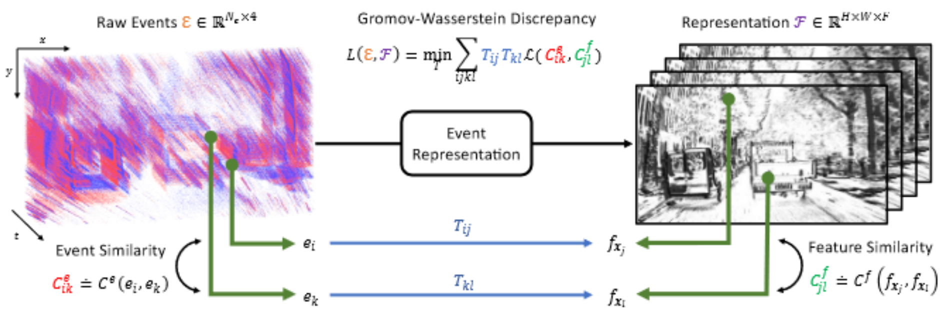
In the study performed to choose the best model to be used, the focus was set on two main approaches. The first model, which currently has the highest mAP, is based on YOLOv6 and integrates the event system using a specific representation: Gromov-Wasserstein Discrepancy. The second approach uses State Space Models (SSM), which have a slightly lower mAP but significantly faster training times. In the following sections, both papers are analyzed to evaluate their strengths and weaknesses and to determine which one is the best.

4.1.1 From Chaos Comes Order: Ordering Event Representations for Object Recognition and Detection

"From Chaos Comes Order" [38] is a paper that focuses on integrating the YOLOv6 architecture with event-based technology. YOLOv6 is an advanced framework for object detection that uses deep neural networks for identifying and classifying objects in images or videos. The effectiveness of YOLOv6 largely depends on the quality of input representations. In the context of event-based computer vision, event representations play a crucial role in determining model performance.

Traditionally, selecting the appropriate representation requires training a neural network for each representation and choosing the best one considering the validation scores. This approach is extremely time-consuming and resource-intensive. To overcome this limitation, the paper proposed using Gromov-Wasserstein Discrepancy (GWD) as a metric for selecting event representations about 200 times faster.

Gromov-Wasserstein Discrepancy



Raw Events $\mathcal{E} \in \mathbb{R}^{N_e \times 4}$
Gromov-Wasserstein Discrepancy
$L(\mathcal{E},\mathcal{F}) = \min_T \sum_{ijkl} T_{ij} T_{kl} \mathcal{L}(C^{\mathcal{E}}_{ik}, C^{f}_{jl})$
Representation $\mathcal{F} \in \mathbb{R}^{H \times W \times F}$
x
y
t
Event Representation
e_i
e_k
T_{ij}
T_{kl}
f_{x_j}
f_{x_l}
Event Similarity
$C^{\mathcal{E}}_{ik} \doteq C^{e}(e_i, e_k)$
Feature Similarity
$C^{f}_{jl} \doteq C^{f}(f_{x_j}, f_{x_l})$

Figure 4.1: Overview of the Gromov-Wasserstein Discrepancy (GWD) [38]

The Gromov-Wasserstein Discrepancy (GWD) measures the distortion introduced in converting raw events into dense representations. Formally, the GWD between a set of events $\mathscr{E}$ and their representations $\mathscr{F}$ is defined as the solution to an optimal transport problem:

$$L(\mathscr{E},\mathscr{F}) = \min_T \sum_{i,j,k,l} T_{ij} T_{kl} L(C^{\mathscr{E}}_{ik}, C^{\mathscr{F}}_{jl}), \tag{4.1.5}$$

where T is the transport plan that moves pairs of events (e_i, e_k) to pairs of features (f_j, f_l), preserving the similarities $C_{ik}^{\mathscr{E}}$ and $C_{jl}^{\mathscr{F}}$ between events and features, respectively. The distortion function $L(C_{ik}^{\mathscr{E}}, C_{jl}^{\mathscr{F}})$ is typically chosen as the Kullback-Leibler Divergence:

$$L(C_{ik}^{\mathscr{E}}, C_{jl}^{\mathscr{F}}) = C_{ik}^{\mathscr{E}} \log \left(\frac{C_{ik}^{\mathscr{E}}}{C_{jl}^{\mathscr{F}}} \right). \tag{4.1.6}$$

The GWD metric can be efficiently calculated and preserves the performance ranking of event representations on validation tasks, making it a preferable alternative to full neural network training for each candidate representation.

Optimizing Representations

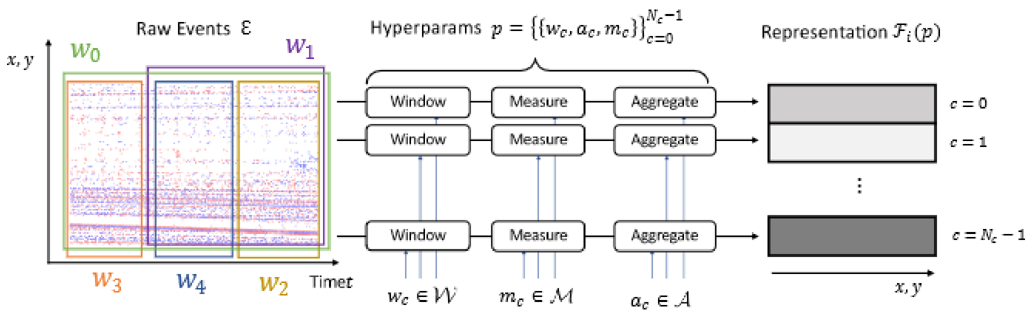


Figure 4.2: Overview of the hyperparameters used to construct an event representation (right) [38].

Using GWD allows a hyperparameter search on a vast family of event representations, revealing new powerful representations that outperform the state-of-the-art. The optimized representation, named ERGO-12 (12-channel Event Representation through Gromov-Wasserstein Optimization), consists of 12 channels selected using Bayesian optimization of the following hyperparameters: measure functions m, aggregation functions a, and time windows w. Formally, each channel c of the representation is defined as:

$$R_c = a_c(m_c(w_c(\mathscr{E}))), \tag{4.1.7}$$

where w_c is the windowing function that selects events within a time interval, m_c is the measure function that selects an event feature (e.g., polarity or timestamp), and a_c is the aggregation function that aggregates the measures into a single feature map.

Integrating Event Representations with YOLOv6

In the paper, the event representations optimized through GWD were integrated into the YOLOv6 framework to improve object detection performance. The integration pipeline is described as follows:

- **Event Pre-processing:** The events captured by the camera are converted into dense representations using the GWD metric. This conversion reduces distortion and preserves crucial information from the original events.
- **Data Ingestion into YOLOv6:** The dense event representations are then used as input to the YOLOv6 network. The network was modified to accept the variable number of channels in the event representations, for example, replacing the 3-channel input convolution with a 12-channel convolution (Nc = 12).
- **Training and Validation:** YOLOv6 is trained and validated using these optimized event representations, demonstrating significant performance improvements over traditional representations.

Experimental Results

Experiments conducted on object detection datasets, such as Gen1 and 1 Mpx, demonstrated that ERGO-12 representations optimized through GWD improved YOLOv6 performance.

This integration represents a significant advancement in event-based computer vision, allowing to fully exploit the powerful detection capabilities of YOLOv6, made of efficiently optimized event representations.

Comparison of Results The results present a comparison between the ERGO-12 method and other state-of-the-art techniques for the Gen1 and 1 Mpx datasets. Although other models may show a higher mAP on the Mpx1 dataset, the importance of the increase with Events+YOLOv3 lies in the specific progress achieved with this combination and the methodology used to achieve these results.

Gen1 Dataset

- **Best Model**: YOLOv6 with SwinV2 backbone and ERGO-12 representation with data augmentation.

- **Achieved mAP**: 0.504.
- **Second Best Model**: RVT-B, with a mAP of 0.472.
- **ERGO-12 without Augmentation**: mAP of 0.493, improving by 2.1% compared to RVT-B.

1 Mpx Dataset

- **Best Model**: YOLOv6 with ERGO-12 and data augmentation.
- **Achieved mAP**: 0.406.
- **Second Best Model**: ASTMNet with a mAP of 0.483 (recurrent).
- **Runner-Up Feed-Forward Method**: Events+YOLOv3, with a mAP of 0.346.
- **ERGO-12 without Augmentation**: mAP of 0.406, improving by 6.0

Summary Table The results of the various models can be summarized in the following table:

Model	Gen1 (mAP)	1 Mpx (mAP)	Transformer
YOLOv6 + ERGO-12 (with Augmentation)	**0.504**	0.406	No
YOLOv6 + ERGO-12 (without Augmentation)	0.493	0.406	No
RVT-B	0.472	0.474	Yes
ASTMNet	0.467	**0.483**	Yes
Events+YOLOv3	0.312	0.346	No

Table 4.1: Performance comparison of various models on Gen1 and 1 Mpx datasets, indicating the use of transformers.

4.1.2 State Space Models for Event Cameras

State Space Models (SSM) are a class of mathematical models particularly effective for handling data from event cameras. These models provide a flexible structure for representing dynamic systems, allowing efficient temporal aggregation and adaptability to different inference frequencies.

Definition of State Space Models

A continuous linear state space model is described by the following equations:

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \tag{4.1.8}$$

$$y(t) = Cx(t) + Du(t) \tag{4.1.9}$$

Where:

- $x(t) \in \mathbb{R}^P$ is the latent state vector.
- $u(t) \in \mathbb{R}^U$ is the input vector.
- $y(t) \in \mathbb{R}^M$ is the output vector.
- $A \in \mathbb{R}^{P \times P}$ is the state transition matrix.
- $B \in \mathbb{R}^{P \times U}$ is the input matrix.
- $C \in \mathbb{R}^{M \times P}$ is the output matrix.
- $D \in \mathbb{R}^{M \times U}$ is the direct transmission matrix.

Each of these components is analyzed in detail:

Latent State Vector $x(t)$ The latent state vector $x(t)$ represents the internal state of the system at a given time. This vector contains all the information needed to fully describe the system's state, evolving over time according to the dynamics defined by the state transition matrix A.

Input Vector $u(t)$ The input vector $u(t)$ represents external influences acting on the system. These influences can include control variables, disturbance signals, or other external forces affecting the system's dynamics. The input matrix B determines how these external inputs modify the system's state.

Output Vector $y(t)$ The output vector $y(t)$ represents the observations or measurements of the system that can be obtained. This vector is a function of the latent state $x(t)$ and the input $u(t)$, calculated through the output matrix C and the direct transmission matrix D.

State Transition Matrix A The state transition matrix A defines how the latent state $x(t)$ evolves over time. It determines the relationship between the current state and the rate of change of the state, i.e., how the dynamic system responds to changes in the state itself.

Input Matrix B The input matrix B specifies how the input vectors $u(t)$ influence the rate of change of the latent state $x(t)$. This matrix is essential for incorporating the effects of external inputs into the system's dynamics.

Output Matrix C The output matrix C links the latent state $x(t)$ to the observations $y(t)$. It defines how the system's internal state is mapped to observable measurements, allowing the interpretation of the system's behavior through its outputs.

Direct Transmission Matrix D The direct transmission matrix D describes the direct relationship between the input $u(t)$ and the output $y(t)$. In many cases, this matrix can be zero, indicating no direct transmission between input and output without passing through the latent state.

Model Discretization

To make the model practical on modern hardware, given a fixed step size Δ, this continuous-time model can be discretized into a linear recurrence using various methods such as Euler, bilinear, or zero-order hold (ZOH):

$$x_k = \overline{A}x_{k-1} + \overline{B}u_k, \quad y_k = \overline{C}x_k + \overline{D}u_k$$

The parameters in the discrete-time model are functions of the continuous-time parameters, defined by the chosen discretization method.

Model Structure

A hierarchical 4-stage backbone has been designed. Each stage uses the features processed by the previous stage and reuses the SSM (State Space Model) state from the last timestep to compute new features. By saving the SSM states, each stage preserves temporal information for

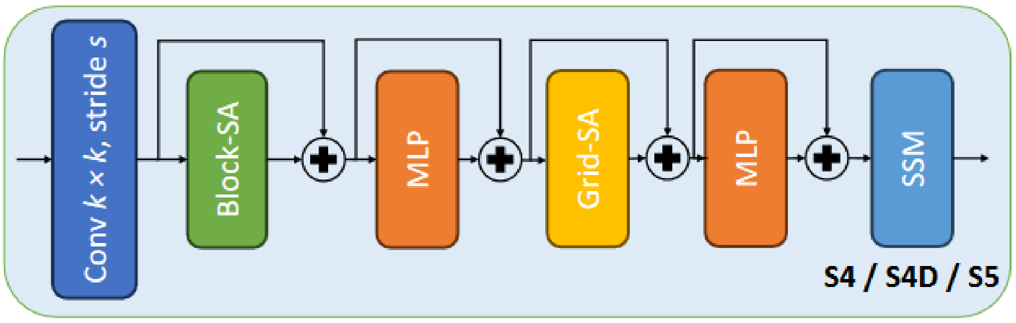


Figure 4.3: SSM-ViT block structure [43].

the entire feature map, allowing generalization to different frequencies thanks to the use of the SSM model.

In the block structure, the initial input undergoes a convolution with a k×k kernel and a stride s. This operation effectively captures the essential spatial features of the input. After the convolution, the structure introduces a module called 'Block-SA'. This module implements self-attention mechanisms, but it does this specifically within local windows. The localized attention in this block ensures a focus on immediate spatial relationships, allowing a detailed representation of close proximity features.

Following 'Block-SA', the 'Grid-SA' module comes into play. Unlike the previous block, 'Grid-SA' uses dilated attention, offering a global perspective. This module, considering a wider scope, provides a more comprehensive understanding of the spatial relationships and the overall structure of the input.

4.1.3 Why State Space Models for Event Cameras are Better than From Chaos Comes Order

Method	**Gen1 mAP**	**Gen1 Time (ms)**	**1 Mpx mAP**	**1 Mpx Time (ms)**	**Params (M)**
ERGO-12	**50.4**	69.9	40.6	100.0	59.6
S4D-ViT-B	46.2	9.40	46.8	10.9	16.5
S5-ViT-B	47.4	**8.16**	47.2	**9.57**	18.2

Table 4.2: Comparisons on test sets of Gen1 and 1 Mpx datasets (20 Hz). Best results in bold and second best underlined.

SSM models, such as the previously mentioned ERGO-12, have been evaluated on event camera datasets, Gen1 and 1 Mpx, demonstrating superior performance compared to existing methods. Experimental results indicate that SSM models train 33% faster and exhibit minimal performance degradation during inference, with a reduction of only 3.76 mAP points.

"State Space Models for Event Cameras" offers, compared to GWD-based representations introduced in "From Chaos Comes Order", significant advantages in terms of training speed and generalization capability As highlighted in the table 4.2, SSM models ensure considerably higher inference speeds.

4.2 Event-based SLAM Framework Used in the Project

Choosing an event-based SLAM framework was a significant challenge due to the relative novelty and limited use of event sensors.There are two main approaches: Graph-Based SLAM and Filter-Based SLAM. Additionally, for event processing and feature extraction, there are two methodologies: direct feature extraction (corners and edges) from events, and converting events into frames followed by feature extraction using standard algorithms (for more details, refer to Appendix A).

The choice fell on Graph-Based SLAM. This decision was influenced by the fact that the University of Luxembourg's department uses this type of SLAM due to its high scalability. Regarding the processing methodology, it would have been preferable to use direct feature extraction from events, as this solution is more computationally lightweight and certainly more innovative. Unfortunately, it was not possible to adopt this solution because the frameworks discussed in various research articles were not open-source. Unfortunately, it was not possible to adopt this solution due to the discussed frameworks being not open-source.

The framework considered was the one discussed in the Ultimate SLAM paper [21]. Below, this paper will be explained in detail.

4.2.1 State Estimation Pipeline with Hybrid Sensors on Ultimate SLAM

The state estimation pipeline presented in this work combines three different types of sensors(see Figure 4.4): event sensors, standard images, and inertial measurement units (IMU). This combination leverages the complementary advantages of each sensor, significantly improving the accuracy and robustness of state estimation in high-speed and high dynamic range

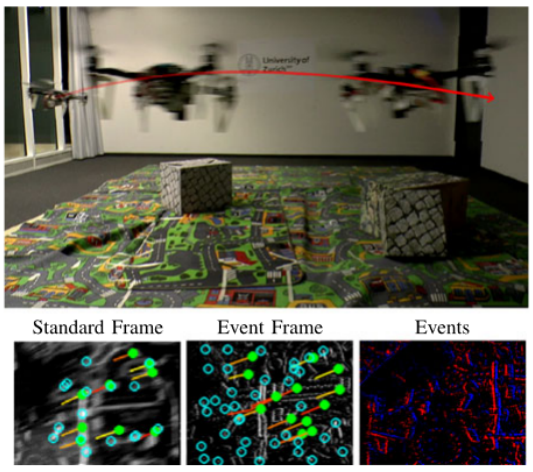


Figure 4.4: Bottom Left: Standard frame, Bottom Middle: Virtual event frame, Bottom Right: Events only (blue: positive events, red: negative events) [21].

scenarios. The pipeline is described in the following steps:

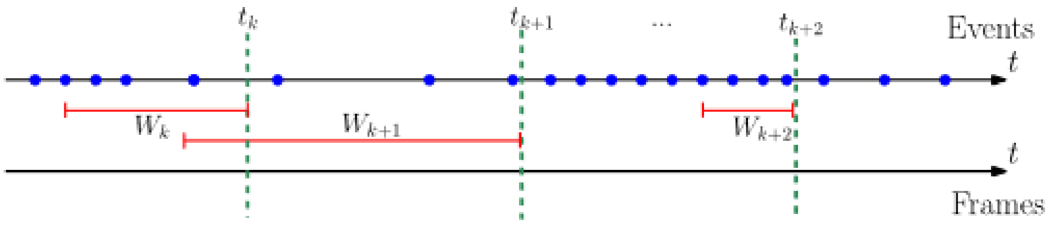


Figure 4.5: Upon receiving a new frame from the standard camera at time t_k, a spatio-temporal window of events W_k is selected, containing a fixed number of events ($N = 4$ in this example). Note that the temporal size of each window is automatically adapted to the event rate. Blue dots correspond to events, and the dashed green lines correspond to the times in which standard frames are received. The bounds of the spatio-temporal windows of events considered are marked in red [21].

Synchronization of Event Spatio-Temporal Windows The spatio-temporal windows of events are synchronized with the timestamps of standard frames. Upon receiving each standard frame at time t_k, a new spatio-temporal event window W_k is created:

$$W_k = \{e_j(t_k - N + 1), \dots, e_j(t_k)\}$$

where $j(t_k)$ is the index of the event with timestamp $t_j < t_k$, and N is the window size (see Figure

4.5).

Synthesis of Motion-Compensated Event Frames Each spatio-temporal event window is synthesized into an event frame I_k by correcting the motion of each event based on its timestamp:

$$I_k(x) = \sum_{e_i \in W_k} \delta(x - x_i')$$

where $\delta(x)$ is the Kronecker delta function, and x_i' is the corrected position of the event obtained by:

$$x_i' = \pi_0(T_{t_k,t_i}(Z(x_i)\pi_0^{-1}(x_i)))$$

Here, $\pi_0(.)$ represents the event camera projection model, T_{t_l,t_m} is the incremental transformation between camera poses at times t_l and t_m, and $Z(x_i)$ is the depth of the scene at time t_i and pixel x_i.

Feature Tracking Features are extracted using the *FAST corner detector* on both the synthesized event frames and the standard frames. These features are independently tracked using the *KLT tracker*, generating two sets of feature tracks:

$$\{z_{0,j,k}\}, \{z_{1,j,k}\}$$

where j is the feature track index and k is the frame index.

Visual-Inertial Fusion through Non-Linear Optimization The visual-inertial localization and mapping problem is formulated as the joint optimization of a cost function that includes three terms: two weighted reprojection errors (one for event camera observations and one for standard camera observations) and an inertial error term:

$$J = \sum_{i=0}^{1} \sum_{k=1}^{K} \sum_{j \in J(i,k)} e_{i,j,k}^T W_{i,j,k} e_{i,j,k} + \sum_{k=1}^{K-1} e_k^T W_k e_k$$

where i denotes the sensor index, k denotes the frame index, and j denotes the landmark index. The reprojection error is defined as:

$$e_{i,j,k} = z_{i,j,k} - \pi_i(T_{C_iS}T_{SW}l_{i,j})$$

The optimization is performed over a limited set of frames, consisting of M keyframes and a sliding window containing the last K frames, using the Google Ceres optimizer.

4.2.2 Results

The implementation of the proposed pipeline on an autonomous quadcopter demonstrated its ability to operate in challenging scenarios, such as low-light environments and high-speed movements, with a 130% improvement in accuracy compared to using events and IMU alone, and an 85% improvement compared to using standard frames and IMU alone.

Sequence	**Proposed (Fr + E + I)**		**E + I**		**Fr + I**	
	Position Error (%)	**Yaw Error (deg/m)**	**Position Error (%)**	**Yaw Error (deg/m)**	**Position Error (%)**	**Yaw Error (deg/m)**
boxes_6dof	**0.30**	**0.04**	0.44	0.05	0.30	0.06
boxes_translation	0.27	**0.02**	0.76	0.05	**0.17**	0.03
dynamic_6dof	**0.19**	0.10	0.38	**0.06**	0.62	0.10
dynamic_translation	**0.18**	**0.15**	0.59	0.16	0.67	0.26
hdr_boxes	**0.37**	**0.03**	0.67	0.09	0.78	0.17
hdr_poster	0.31	0.05	0.49	**0.04**	**0.28**	0.08
poster_6dof	0.28	0.07	0.30	0.08	0.59	0.11
poster_translation	**0.12**	0.04	0.15	**0.04**	0.23	0.08
shapes_6dof	**0.10**	**0.04**	0.48	0.06	0.17	0.05
shapes_translation	**0.26**	0.06	0.41	**0.04**	0.29	0.11

Table 4.3: Comparison between Proposed (Fr + E + I), (E + I), and (Fr + I)

CHAPTER 5

Proposed Methodologies: Integrating Ultimate SLAM and SSM Frameworks

This section presents the proposed methodologies for integrating Ultimate SLAM and State Space Models (SSM). The aim is to create a unified framework that leverages the strengths of both approaches. The proposed methods include offline and online implementations, each tailored to address specific challenges in SLAM and object detection. Detailed descriptions of the algorithms, implementation steps, and theoretical underpinnings will be provided.

5.1 Offline Implementation

This was the first implementation using the SSM framework in offline mode. The attached figure 5.1 provides an overview of the workflow, described in the following detailed steps:

1. *Time t0*: Open the `.bag` file and convert it entirely into the raw format required by SSM, which will be preprocessed later.
2. *Time t1*: After conversion, perform inference on the extracted events and obtain the predicted bounding boxes.
3. *Time t2*: Replay the same `.bag` file used at time t0 using the `rosbag play` command. Send the data to an Ultimate SLAM ROS node, which uses the acquired events to emulate a real-time system, extract the poses, and simultaneously apply the previously obtained bounding boxes to the event frames.

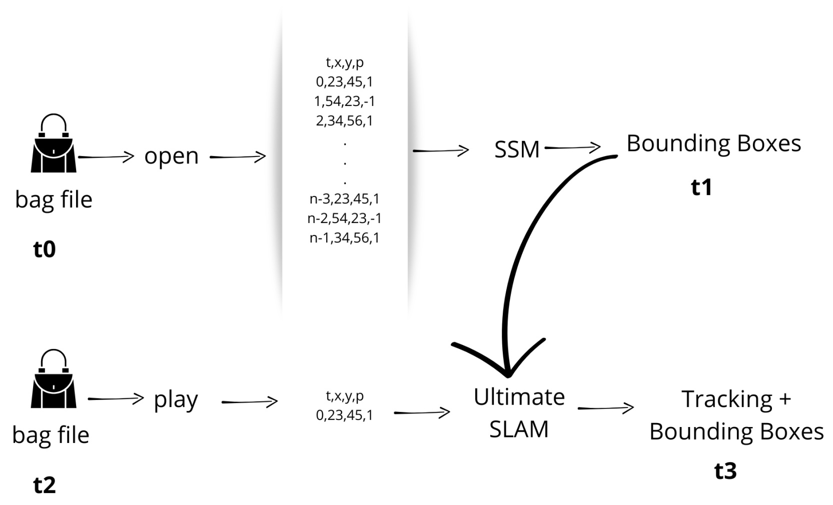


Figure 5.1: Offline Implementation Flow

4. *Time t3*: Using a ROS node, send bounding boxes, poses, and event frames in a synchronized manner as topics, making them available for further processing by other ROS nodes.

This preliminary implementation was useful for verifying the feasibility of the entire process. However, it is impractical for real-world applications. In a real scenario, recording an environment with an event camera and then reprocessing the information with the bounding boxes would be inefficient. This process requires a complete recording before the data can be processed, which is not sustainable in an automated or robotic system where real-time processing is required. For a truly operational system, an approach that integrates data collection and processing in real-time is necessary.

5.2 Online Implementation

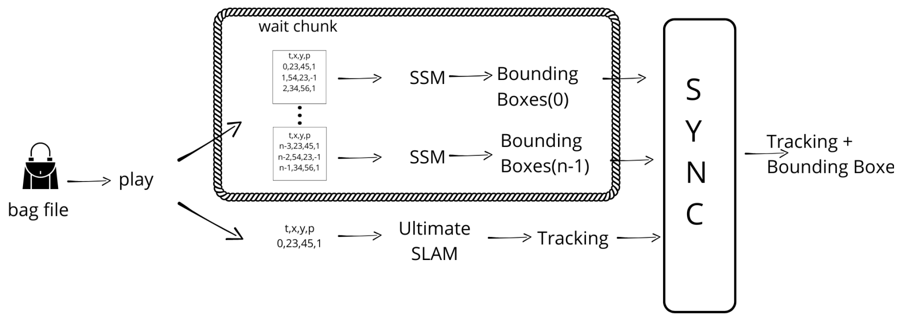


Figure 5.2: Online Implementation Flow

After verifying the feasibility of the offline implementation, it was necessary to develop a version that works in real-time or near real-time. The steps followed are described below and shown in the attached figure 5.2.

Creation of ROS Nodes Instead of reading the whole `.bag` file like a normal file, a ROS node was created. This node subscribes to the `events` topic in the `.bag` file. In this way, SSM reads only part of the `.bag` file at a time, processing the events in small blocks.

Thanks to this contribution, SSM can now be used by anyone with the standard robotic framework. Before this implementation, SSM did not offer the ability to communicate directly with a robotic system, limiting its accessibility and integration. With the creation of ROS nodes, a smooth and continuous communication between SSM and robotic systems was enabled, making it easier to use SSM in robotic applications. This progress allows developers and researchers to easily integrate SSM into their projects, fully using the potential of the ROS framework for robot control and management. The importance of ROS lies in its widespread use as the de facto standard for robotics, and enabling SSM compatibility with ROS means making this powerful tool accessible to a much larger user community.

Initial Processing and Parallelization of Processes The ROS node reads an initial portion of the `.bag` file and processes it to generate the first bounding boxes, which are sent to a synchronization node. Meanwhile, the next chunks are preprocessed and detected. This allows for quick initial results without waiting for the entire file to be read.

Running Ultimate SLAM in Parallel To make the entire system work in online mode, the Ultimate SLAM framework is run simultaneously.

Role of the Synchronizer The role of the synchronizer is crucial. It uses the timestamps of the event frames produced by Ultimate SLAM and the timestamps of the bounding boxes to synchronize everything. The synchronization involves storing Ultimate SLAM information in memory while waiting for the bounding boxes. This process depends on the computing power and the system's ability to handle multiple processes. After an initial delay Δt, thanks to the parallelization of the inference processes, we will be able to send synchronized data.

CHAPTER 6

Validation and Experimental Results

6.1 Thesis Objectives

The main goal of this thesis is to develop a solution for integrating the SLAM (Simultaneous Localization and Mapping) system with object detection within the ROS (Robot Operating System) environment. This integration is crucial to adapt the system to a robotic context, enabling accurate reconstruction of the surrounding environment.

Available Tools

- *DAVIS 240C Datasets*: These datasets contain monocular videos captured with the DAVIS 240 camera, providing essential information for processing.

Specific Objectives

1. *Determining the 6D Pose of the Sensor Relative to the Environment*:
    - Identify the position and orientation of the sensor in three-dimensional space.
2. *Determining the 6D Pose of Objects Relative to the Sensor*:
    - Detect and track the position and orientation of detected objects relative to the sensor.
3. *Integrating the System into ROS*:
    - Develop an integrated system that uses ROS for online processing of sensor information and implementation in a robotic context.

The 6D poses (6-DOF, Degrees of Freedom) represent the position and orientation of an object in three-dimensional space. A description of the 6D poses includes six components: three for position and three for orientation.

- **Position (Translation)**:
  - **X**: Coordinate along the X-axis.
  - **Y**: Coordinate along the Y-axis.
  - **Z**: Coordinate along the Z-axis.
- **Orientation (Rotation)**:
  - **Roll**: Rotation around the X-axis.
  - **Pitch**: Rotation around the Y-axis.
  - **Yaw**: Rotation around the Z-axis.

These objectives are essential for ensuring an accurate reconstruction of the environment and improving the navigation and interaction capabilities of a robotic system.

6.2 Description of the Validation Dataset

As mentioned in the introductory part of the chapter, the DAVIS 240C dataset was used. This dataset provides `.bag` files containing events. It is known for its event videos that show sudden and rapid camera movements, demonstrating how efficiently a neuromorphic camera's events can capture environmental features. Among all the available files, the `dynamic_rotation.bag` file was selected because it had a more moderate speed, allowing more accurate experiments using the ground truth to monitor sensor poses. Additionally, it included the presence of a person, an essential element for conducting object detection experiments.

6.3 Evaluation Metrics Used

Several metrics were used to evaluate the system's performance:

- Quantitative

- Qualitative

The choice of metrics depends on the availability of ground truth data for the sensor poses.

To evaluate the accuracy of the estimated poses compared to the ground truth, the following metrics were calculated:

- *Absolute Position Error (APE)*: The absolute error measures the difference between the system's estimated position and the actual position in terms of translation and rotation. The formula to calculate APE is:

  $$\mathrm{APE}(t) = \left\| \mathbf{p}_{\mathrm{gt}}(t) - \mathbf{p}_{\mathrm{est}}(t) \right\|$$

  where $\mathbf{p}_{\mathrm{gt}}(t)$ is the ground truth position at time t and $\mathbf{p}_{\mathrm{est}}(t)$ is the estimated position at time t.

- *Relative Position Error (RPE)*: The relative error evaluates the consistency of the estimated trajectory by measuring the difference between successive relative transformations in the estimated and actual trajectories. The formula to calculate RPE is:

  $$\mathrm{RPE}(t, \Delta t) = \left\| (\mathbf{p}_{\mathrm{gt}}(t+\Delta t) - \mathbf{p}_{\mathrm{gt}}(t)) - (\mathbf{p}_{\mathrm{est}}(t+\Delta t) - \mathbf{p}_{\mathrm{est}}(t)) \right\|$$

  where $\mathbf{p}_{\mathrm{gt}}(t)$ and $\mathbf{p}_{\mathrm{gt}}(t+\Delta t)$ are the ground truth positions at times t and $t+\Delta t$, and $\mathbf{p}_{\mathrm{est}}(t)$ and $\mathbf{p}_{\mathrm{est}}(t+\Delta t)$ are the estimated positions at times t and $t+\Delta t$.

These metrics help understand how well the estimated poses align with reality.

Using the Umeyama Algorithm allowed to:

- Realign the trajectory
- Reduce the absolute error
- Reduce the relative error
- Significantly improve the accuracy of the estimated poses

The Umeyama algorithm is used for rigid alignment of two sets of points in three-dimensional space, minimizing the mean square error between corresponding points. The optimal transformation $(\mathbf{R}, \mathbf{t}, s)$, where $\mathbf{R}$ is the rotation matrix, $\mathbf{t}$ is the translation vector, and s is the scaling factor, is found by minimizing:

$$\min_{\mathbf{R},\mathbf{t},s} \sum_i \left\| s\mathbf{p}_{\mathrm{gt},i} - (\mathbf{R}\mathbf{p}_{\mathrm{est},i} + \mathbf{t}) \right\|^2$$

The Umeyama algorithm solves this problem using matrix decompositions and sample means.

- *Alignment Rotation*: The rotation matrix **R** is calculated using the singular value decomposition (SVD) of the covariance matrix between corresponding points.
- *Alignment Translation*: The translation vector **t** is determined by the difference between the centroids of the two sets of points.
- *Scale Correction*: The scaling factor s is calculated as the ratio between the sum of squared distances of the points in the reference set and the sum of squared distances of the estimated points after applying rotation and translation.

An example output of the Umeyama algorithm might be:

$$\mathbf{R} = \begin{bmatrix} 0.99 & -0.03 & 0.04 \\ 0.03 & 0.99 & -0.02 \\ -0.04 & 0.02 & 0.99 \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} 0.5 \\ -0.2 \\ 0.1 \end{bmatrix}, \quad s = 1.02$$

For Object Detection, without the ground truth of the bounding boxes extracted from the event videos, a qualitative evaluation was used:

- Visual inspection of detected bounding boxes
- Visual analysis of each video frame to check the accuracy of the bounding boxes
- Identification of any obvious errors in object detection

6.4 Challenges and Solutions Implemented for Ultimate SLAM Adaptation

During the development and setup of the Ultimate SLAM framework, various technical and operational challenges emerged. These challenges are related to:

1. Insufficient documentation.
2. Understanding the correct configuration for the use case.

3. Choosing the most suitable event video from the DAVIS 240C dataset.

Below, each challenge is detailed.

Insufficient Documentation The available documentation did not cover all necessary aspects, making it necessary to study the source code deeply.

Running Ultimate SLAM Ultimate SLAM provides a guide to use the framework in two main modes: "Example" and "Run-Live". The "Example" mode allows the use of available online event videos without needing a physical camera, while the "Run-Live" mode allows use with a real camera.

Since no physical camera was available, the framework was used in "Example" mode to evaluate its behavior. Significant progress was made towards real-time visualization of Ultimate SLAM results. However, a problem was found: the video played at a speed higher than real-time. After some attempts to modify the code to achieve the goal, the complexity of the code was recognized. Changing some instructions to reach real-time speed negatively affected other parts of the code, requiring a complete re-engineering. Given the need for quick results, a more efficient alternative was adopted to complete the remaining steps.

By correctly configuring the "Run-Live" mode configuration file, the system was made able to handle data as if it came from a real camera instead of a `.bag` file. This solved the playback speed problem, allowing real-time result visualization.

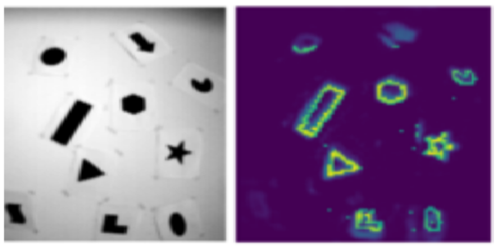
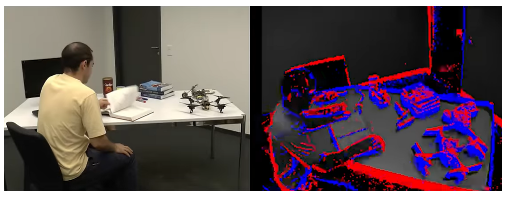



Figure 6.1: On the left, a representation of geometric objects (shapes_rotation.bag), and on the right, a representation of a person moving inside a room (dynamic_translation.bag), both acquired with an event camera.

Appropriate Event Video Many available event videos are designed to show the cameras' capabilities in extreme conditions, making it hard to interpret pose results. On the other hand, event videos with more "common" scenes often lack ground truth.

The ideal compromise was reached with the `dynamic_translation.bag` event video (see Figure 6.1), which shows a person moving from a desk with moderate rotational movements. This event video offers a good balance between suitable playback speed and the presence of ground truth. Although the camera movement was limited to slight shifts, which did not significantly contribute to pose variety, it was still a good starting point.

Additionally, the presence of a person in the event video was particularly useful for integration with object detection algorithms trained on people and vehicles. In contrast, other available event videos showed only geometric shapes, which were not useful for the specific purpose.

6.5 Challenges and Solutions Implemented for SSM Adaptation

During the development and setup of the SSMS framework, various technical and operational challenges emerged. These challenges are related to:

1. Insufficient documentation.
2. Preprocessing of raw data.
3. Adapting bag files to the required format for preprocessing.
4. Inadequacy of the framework for tests on data without ground truth.
5. Unrecognized poses.

Below, each challenge is detailed.

Insufficient Documentation The available documentation did not cover all necessary aspects, making it essential to study the source code.

Preprocessing Raw Data Initially, the documentation helped to understand how to use preprocessed data to train, validate, and test the model. However, it did not clarify how to transform raw data into preprocessed data, as the goal was to use bag files employed in Ultimate SLAM to test the model.

The first test was conducted using preprocessed data from the Gen1 dataset, as according to the table in the document, this dataset showed slightly better performance than Mpx1. After verifying the framework's functionality, raw files were downloaded from the official Gen1 site to understand and ensure that the transformation from raw to preprocessed data matched the following structure:

```
Example structure of a sequence:
 event_representations_v2
    ev_representation_name
        event_representations.h5
        objframe_idx_2_repr_idx.npy
        timestamps_us.npy
 labels_v2
     labels.npz
     timestamps_us.npy
```

This step was crucial to determine the types of raw files needed by the framework before preprocessing.

Converting the bag file into preprocessable files The next challenge was understanding how to use a bag file to convert it into a preprocessable file. The bag file was opened like a normal file, since SSMS does not support ROS nodes, and the data was extracted and transformed into the required format for preprocessing. Examining some types of files is not simple, so many experiments were conducted to test the correctness of the conversion.

Inadequacy of the framework for tests on data without ground truth Once the `.bag` file was converted into a preprocessable format, the next problem was that the framework could not preprocess the file containing the events, as it required a ground truth file. This raises a question: if the model is to be applied in daily life where there is no ground truth, why is it needed? The only answer found is that the University of Zurich, when implementing these frameworks (including past ones), uses classical datasets containing all annotation information to train, validate, and test models, without considering the need to test the models in other contexts. This strategy allows a comparison between frameworks using the same dataset but is not fully usable in an industrial context without the appropriate code modifications.

To overcome this stage, a fake `.npy` ground truth file was created, containing random bounding boxes for the entire duration of the `.bag` video, keeping the maximum resolution size. This file was then passed to the preprocessing process. Even though the framework still required a comparison with the ground truth for validation, once the prediction was obtained, the predicted bounding boxes could be overlaid on the video.

Unrecognized poses One of the goals was to extract the 6D poses of objects. To do this, object detection was used, but this was not enough; depth recognition techniques were needed to extract the 6D poses. This could be achieved using specially designed physical tools or estimating the poses using models trained for this purpose.

In the context of RGB images, there are various models that, given a sequence of bounding boxes, can estimate an object's pose. However, in the context of event cameras, events potentially represent sparse points, making it difficult to estimate depth and, consequently, object poses. Despite the research conducted, there are still not enough studies in this field to provide effective solutions.

6.6 Experimental Results with Ultimate SLAM

1. *Subscription to the Pose Topic:* A node was used to subscribe to the pose topic, which receives the poses generated by the set of event frames, standard frames, and IMU.
2. *Using evo:* The Python package *evo* was used, dedicated to evaluating odometry and SLAM. This tool allows analysis of the trajectories obtained from the system and calculates various types of errors compared to a reference trajectory.
3. *Extraction of Absolute and Relative Errors:*

Analysis of Table 6.1 - Absolute Position Error (APE)

- *Max (m):* The maximum error is slightly higher for the *Event frames + IMU* configuration. This could indicate that, in specific cases, including standard frames helps reduce error spikes.
- *Mean (m):* The absolute mean error is lower for the configuration that includes standard frames, suggesting an overall more accurate estimate.
- *Median (m):* The median error is also lower for the configuration with standard frames, indicating that most errors are smaller compared to the *Event frames + IMU* configuration.

Method	*Event frames + IMU*	*Standard frames + Event frames + IMU*
Absolute Position Error (APE) - Not Aligned		
Max (m)	3.291225	3.042086
Mean (m)	2.691573	2.515195
Median (m)	2.643869	2.490139
Min (m)	2.362331	2.095063
RMSE (m)	2.697540	2.521096
SSE	13149.040140	22741.508319
Std (m)	0.179333	0.172404
Relative Position Error (RPE) - Not Aligned		
Max (m)	0.195109	0.079702
Mean (m)	0.004238	0.002966
Median (m)	0.003015	0.002028
Min (m)	0.000177	0.000067
RMSE (m)	0.007645	0.004898
SSE	0.105550	0.085802
Std (m)	0.006362	0.003898

Table 6.1: Summary of absolute and relative errors for Event frames + IMU and Standard frames + Event frames + IMU of the dynamic_rotation event video.

- *Min (m):* The minimum error is lower in the configuration that includes standard frames, which could indicate greater accuracy in the best cases.
- *RMSE (m):* The root mean square error is lower for the configuration with standard frames, suggesting overall better accuracy.
- *SSE:* The sum of squared errors is significantly higher for the *Standard frames + Event frames + IMU* configuration. This is normal and expected because this configuration can record more poses compared to *Event frames + IMU*, thus increasing the total summed errors.
- *Std (m):* The standard deviation of the error is slightly lower for the configuration with standard frames, indicating a tighter and more consistent error distribution.

Analysis of Table 6.1 - Relative Position Error (RPE)

- *Max (m):* The maximum relative error is much lower for the configuration that includes standard frames, suggesting that transitions between consecutive poses are estimated more accurately.
- *Mean (m):* The relative mean error is lower for the configuration with standard frames, indicating greater consistency in estimated movements.
- *Median (m):* The relative median error is also lower for the configuration with standard frames, further supporting the idea of a more accurate estimation of movements.
- *Min (m):* The minimum relative error is lower for the configuration with standard frames, suggesting that smaller movements are captured more accurately.
- *RMSE (m):* The relative root mean square error is lower for the configuration with standard frames, indicating overall better accuracy in transitions.
- *SSE:* The sum of relative squared errors is slightly lower for the configuration with standard frames, suggesting overall better consistency in estimated movements.
- *Std (m):* The standard deviation of the relative error is lower for the configuration with standard frames, indicating a tighter and more consistent error distribution.

4. *Application of the Umeyama Algorithm*

5. *Recalculation of Absolute and Relative Errors:* After applying the Umeyama algorithm to align the estimated trajectory with the reference trajectory, absolute and relative errors were recalculated. This step is crucial to accurately evaluate the system's performance, as the alignment corrects any discrepancies due to initial position and orientation differences.

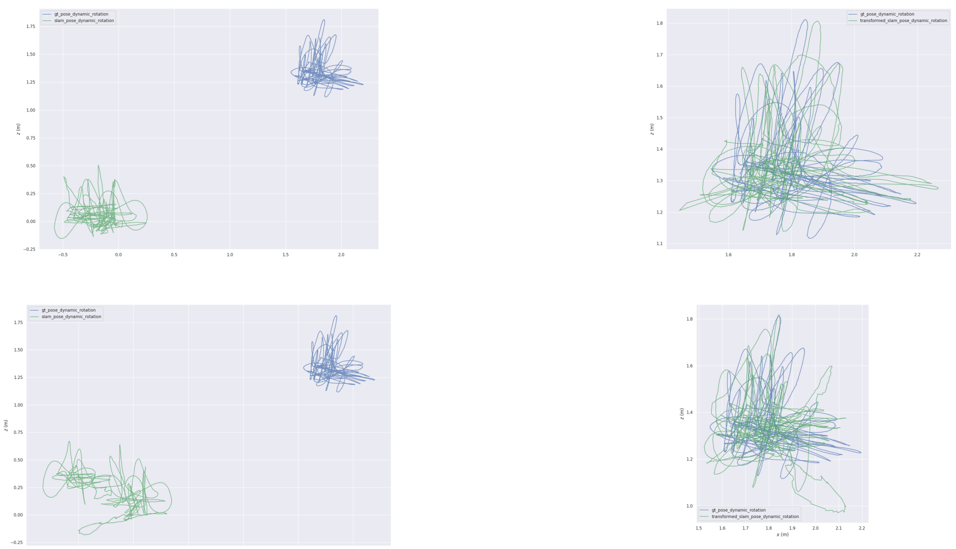


Figure 6.2: In blue the ground truth trajectory, while the green one is calculated by our system. Top left shows the behavior using Event Frames + IMU before applying the Umeyama algorithm. Top right shows the same data but with the Umeyama algorithm applied. Bottom left shows the behavior using Standard Frames + Event Frames + IMU before applying the Umeyama algorithm. Bottom right shows the same data but with the Umeyama algorithm applied.

Using the alignment, as seen in Table 6.2, there is a substantial improvement in values, but the result still indicates that using event cameras together with the standard camera remains advantageous.

According to the Ultimate SLAM article [21], which proposes the combined use of standard frames, event frames, and IMU, and based on the analyses, it can be stated that the simultaneous use of both types of sensors, namely the standard camera and the event camera, creates an unprecedented complementarity. Depending on the situations, one of the two sensors can certainly be more advantageous than the other: the standard camera is ideal in normal light conditions as it captures more details, while the event camera is more effective in extreme or inadequate lighting conditions for the RGB camera. Therefore, the choice of the sensor depends on the specific application, and all three strategies can be valid.

The dataset used is not ideal for this type of analysis, as it includes high-speed rotations to demonstrate the importance of the event camera, capable of detecting angles at speeds that the RGB camera cannot handle due to motion blur. However, since visualizing the trajectory can be problematic, a more detailed approach could be adopted to examine displacement over time rather than space. By analyzing the ground truth and the estimate by observing the displacement

Method	*Event Frames + IMU*	*Standard Frames + Event Frames + IMU*
Absolute Position Error (APE) - Aligned		
Max (m)	0.355371	0.805634
Mean (m)	0.121434	0.406349
Median (m)	0.110556	0.414892
Min (m)	0.016400	0.128528
RMSE (m)	0.136971	0.431321
SSE	33.901328	665.641938
Std (m)	0.063364	0.144630
Relative Position Error (RPE) - Aligned		
Max (m)	0.201796	0.085581
Mean (m)	0.020917	0.012095
Median (m)	0.016805	0.010790
Min (m)	0.000271	0.000494
RMSE (m)	0.027623	0.015079
SSE	1.378049	0.813332
Std (m)	0.018042	0.009005

Table 6.2: Summary of aligned absolute and relative errors for Event Frames + IMU and Standard Frames + Event Frames + IMU after using the Umeyama alignment on the dynamic_rotation event video.

over time, a clearer view of the movement and discrepancies between the two datasets can be obtained. As seen in Figure 6.3, especially after using the Umeyama algorithm, the system's time series is very similar to the ground truth.

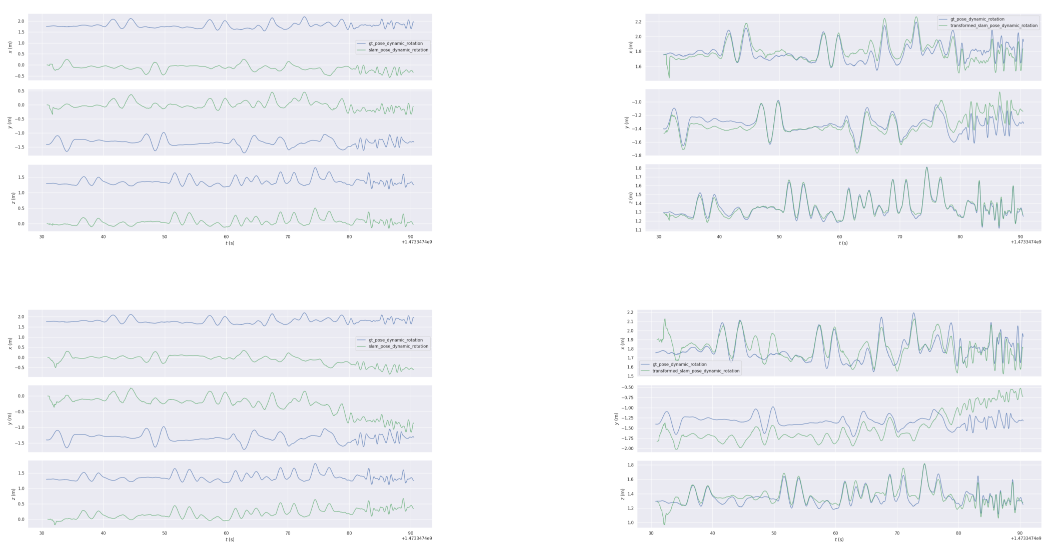


Figure 6.3: In blue the time series of the ground truth, while the green one is calculated by our system. Top left shows the behavior using Event Frames + IMU before applying the Umeyama algorithm. Top right shows the same data but with the Umeyama algorithm applied. Bottom left shows the behavior using Standard Frames + Event Frames + IMU before applying the Umeyama algorithm. Bottom right shows the same data but with the Umeyama algorithm applied.

6.7 Experimental Results with SSM

Regarding the bounding box results using the event video "dynamic_rotation" also used for poses, it emerged that the model, being trained in outdoor environments, showed confusion when operating indoors. This caused the incorrect recognition of some unknown objects as one of the two targets it was trained for, namely pedestrians and cars.

For example, as shown in the attached figures, the model can recognize a person but struggles to correctly identify objects like computers and desks, confusing them with vehicles or pedestrians. A clear example highlighting the model's limitations is the failure to recognize the legs of a sitting person. This happens because the model was trained with recordings from an event camera capturing a moving street, with people walking and cars moving. As a result, the model has never seen a sitting person, only standing.

It should also be noted that despite these limitations, the model is one of the best currently available. However, it still has only 47% mAP. A value of 47% indicates that while the model can correctly identify a good portion of objects (outdoors), there is still ample room for improvement in both precision and detection coverage.

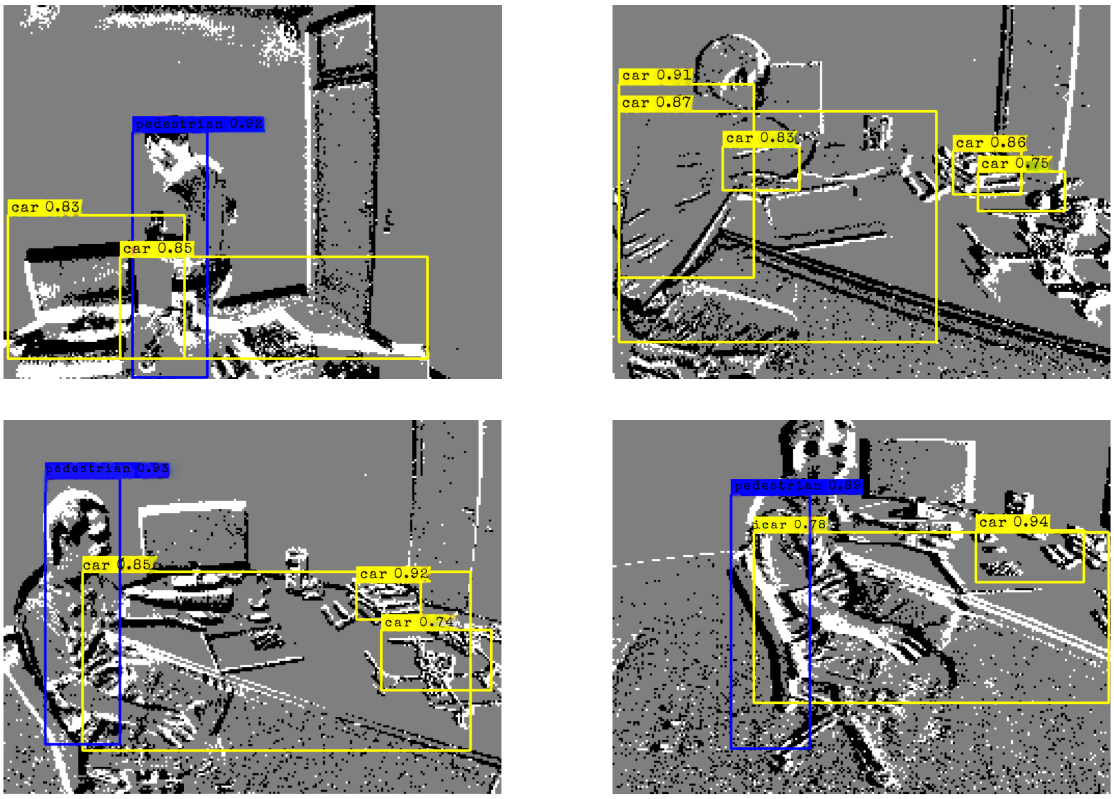


Figure 6.4: Examples of object detection on the dynamic_rotation event video

To improve the model's performance in environments different from those it was trained in, it would be useful to integrate the training dataset images with indoor contexts containing various objects. Additionally, it might be beneficial to include scenes showing people in different positions and situations, such as sitting or moving inside a building. This approach would allow the model to develop a greater ability to generalize and adapt better to new scenarios, thus improving its object recognition accuracy.

6.8 Design Discussion

In this section, the obtained results will be discussed compactly, considering the metrics and the achievement of our objectives.

The objectives discussed in section 6.1 were:

1. *Determination of the Sensor's 6D Pose Relative to the Environment*

2. *Determination of the Objects' 6D Pose Relative to the Sensor*

3. *System Integration in ROS*

Based on the modifications made to the frameworks, the results obtained, and following the metrics established in section 6.3, the strengths and limitations encountered in the project will be discussed.

6.8.1 Strengths

- *Determination of the Sensor's 6D Pose Relative to the Environment*:
  - *Pose Optimization*: Significant improvements were made to the poses extracted from the framework by reducing the evident error in trajectory and temporal signal representation. The use of the Umeyama algorithm brought the results closer to the ground truth.
  - *Increased Accuracy*: The combined use of multiple sensors, particularly event frames and standard frames, increased the accuracy of the sensor's 6D poses.
- *System Integration in ROS*:
  - *Dynamic Processing*: Although the framework was designed to process static data already recorded, we managed to make it dynamic. Using integration with ROS, we processed data by passing chunks of events instead of entire blocks.
  - *Online Inference*: This approach allowed us to make inferences on each chunk, managing the system to find bounding boxes during recording.

6.8.2 Limitations

- *Determination of Objects' 6D Pose Relative to the Sensor*:
  - *Search Space Limitations*: We encountered difficulties due to the still limited search space and the complexity in interpreting events, making it impossible to reconstruct the 6D position of objects.
- *False Positives*:
  - *Model Training Problems*: The model was trained on outdoor objects, while our video was indoor, causing false positives. This issue, however, is solvable by training the model with appropriate data.

CHAPTER 7

Conclusions and Future Work

In this thesis, the use of neuromorphic cameras for simultaneous localization and mapping (SLAM) and automatic target detection was explored, integrating the two systems through the ROS framework for communication. The objective of the research was to find a SLAM model that represented a good compromise between economic costs for camera acquisition, open source availability of frameworks, and computational power. Ultimately, the Ultimate SLAM framework [27] was selected. Regarding object detection, the focus was on a trade-off between mean average precision (mAP) and inference times, choosing between State Space Models [43] and From Chaos Comes Order [39]. The choice fell on State Space Models because, despite having a slightly lower mean average precision, the inference times are approximately ten times faster.

This integration allows for high adaptability of both systems in the robotic context. Experimental results have demonstrated the potential of these advanced sensors in improving the efficiency and accuracy of object detection and localization operations. The integration of SLAM and object detection using neuromorphic cameras enables a more responsive and efficient perception system. By using the SSM model, it is possible to achieve inferences in reduced times, thus creating a solid foundation for the development of increasingly real-time oriented systems.

Despite the promising results, there are still several challenges to be addressed:

- *6D Position Estimation*: The use of a single camera, instead of a stereo system, makes it difficult to estimate the 6D position of objects in a room due to intrinsic complexity and the fact that this field is still underexplored.
- *Variety of Datasets*: The limited variety of available datasets is a significant limitation.

Some techniques to overcome this limitation were discussed in section 3.

7.1 Conclusions

This thesis thoroughly explored the use of neuromorphic cameras in the context of SLAM and object detection, achieving the following main results:

1. *Comparison with Framework for SLAM*: The comparative evaluation of various open-source frameworks identified Ultimate SLAM as the most suitable for handling event data with high precision and reliability. This framework has demonstrated robust capability in integrating standard frames, event frames, and IMU to improve pose estimation.
2. *Framework for Object Detection*: A second open-source framework capable of detecting objects based on event data from neuromorphic cameras has been identified. This framework demonstrated the model's ability to recognize certain event patterns, although with some limitations related to the predominantly outdoor training dataset.
3. *Datasets*: Various datasets were analyzed, identifying the most representative ones with quality and variety of data for training and validation of SLAM and object detection frameworks. However, the need to expand the datasets with indoor contexts and different situations emerged to improve the models' generalization capability.
4. *Integration in ROS*: An integrated system using ROS was developed, combining the functionalities of SLAM and object detection synchronously, making the system applicable to real robotic scenarios.

7.2 Future Work

The results obtained represent an important step forward, but there are still several future directions to explore to further improve the system:

1. *Dataset Expansion*: It is crucial to create and integrate new datasets that include indoor environments with a greater variety of objects and situations. This will allow training more robust models capable of better generalization in different contexts.

2. *Improvement of Detection Algorithms*: Developing more advanced detection algorithms that can better handle the peculiarities of event data, reducing false positives, and improving the accuracy of object recognition in complex environments.

3. *Optimization of Integration*: Further optimizing the integration of SLAM and object detection frameworks in ROS to improve computational efficiency and reduce processing times, making the system more responsive and usable in real-time applications.

4. *Validation in Real Scenarios*: Expanding the validation of the integrated system in a wider range of real scenarios and online from a real camera, including dynamic and variable environments, to test the robustness and reliability of the system under different operational conditions.

5. *Understanding Event Properties*: Deepening the understanding of event properties to develop models capable of estimating even 6D poses.

Achieving these objectives will open many new opportunities in this research area.

In conclusion, this work contributes to the growing field of event-based vision, providing methods that pave the way for future advancements. The combination of SLAM and object detection with neuromorphic cameras holds great promise for a wide range of applications, from robotics to autonomous driving, and beyond. Continued research and development in this area will undoubtedly lead to even more innovative solutions and improved performance in artificial vision systems.

References

[1] John J Leonard and Hugh F Durrant-Whyte. “Simultaneous map building and localization for an autonomous mobile robot.” In: *IROS*. Vol. 3. 1991, pp. 1442–1447.

[2] J.E. Guivant and E.M. Nebot. “Optimization of the simultaneous localization and map-building algorithm for real-time implementation”. In: *IEEE Transactions on Robotics and Automation* 17.3 (2001), pp. 242–257. DOI: 10.1109/70.938382.

[3] Michael Montemerlo et al. “FastSLAM: A factored solution to the simultaneous localization and mapping problem”. In: *Aaai/iaai* 593598 (2002).

[4] P. Newman et al. “Explore and return: experimental validation of real-time concurrent mapping and localization”. In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*. Vol. 2. 2002, 1802–1809 vol.2. DOI: 10.1109/ROBOT.2002.1014803.

[5] G Lowe. “Sift-the scale invariant feature transform”. In: *Int. J* 2.91-110 (2004), p. 2.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. “Surf: Speeded up robust features”. In: *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer. 2006, pp. 404–417.

[7] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. “A 128$\times$ 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor”. In: *IEEE Journal of Solid-State Circuits* 43.2 (2008), pp. 566–576. DOI: 10.1109/JSSC.2007.914337.

[8] Deepak Geetha Viswanathan. “Features from accelerated segment test (fast)”. In: *Proceedings of the 10th workshop on image analysis for multimedia interactive services, London, UK*. 2009, pp. 6–8.

[9] Christoph Posch et al. “Retinomorphic Event-Based Vision Sensors: Cameras, Transceivers, and Beyond”. In: *Procedia Computer Science* 7 (2011), pp. 153–158.

[10] Ethan Rublee et al. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International Conference on Computer Vision*. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[11] Simon Ruffieux et al. “A survey of datasets for human gesture recognition”. In: *Human-Computer Interaction. Advanced Interaction Modalities and Techniques: 16th International Conference, HCI International 2014, Heraklion, Crete, Greece, June 22-27, 2014, Proceedings, Part II 16*. Springer. 2014, pp. 337–348.

[12] Garrick Orchard et al. “Converting static image datasets to spiking neuromorphic datasets using saccades”. In: *Frontiers in neuroscience* 9 (2015), p. 159859.

[13] Xingjian Shi et al. “Convolutional LSTM network: A machine learning approach for precipitation nowcasting”. In: *Advances in neural information processing systems* 28 (2015).

[14] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2016, pp. 21–37. ISBN: 9783319464480. DOI: 10.1007/978-3-319-46448-0_2. URL: http://dx.doi.org/10.1007/978-3-319-46448-0_2.

[15] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV].

[16] Shaoqing Ren et al. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. arXiv: 1506.01497 [cs.CV].

[17] Arnon Amir et al. “A low power, fully event-based gesture recognition system”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7243–7252.

[18] Jonathan Binas et al. *DDD17: End-To-End DAVIS Driving Dataset*. 2017. arXiv: 1711.01458 [cs.CV].

[19] Zhen Xie, Shengyong Chen, and Garrick Orchard. “Event-based stereo depth estimation using belief propagation”. In: *Frontiers in neuroscience* 11 (2017), p. 535.

[20] Amos Sironi et al. “HATS: Histograms of averaged time surfaces for robust event-based object classification”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1731–1740.

[21] Antoni Rosinol Vidal et al. “Ultimate SLAM? Combining Events, Images, and IMU for Robust Visual SLAM in HDR and High-Speed Scenarios”. In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 994–1001. DOI: 10.1109/LRA.2018.2793357.

[22] Nabeel Khan and Maria G Martini. “Bandwidth modeling of silicon retinas for next generation visual sensor networks”. In: *Sensors* 19.8 (2019), p. 1751.

[23] Anton Mitrokhin et al. *EV-IMO: Motion Segmentation Dataset and Learning Pipeline for Event Cameras*. 2019.

[24] Liyuan Pan et al. “Bringing a blurry frame alive at high frame-rate with an event camera”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6820–6829.

[25] Pierre De Tournemire et al. “A large scale event-based detection dataset for automotive”. In: *arXiv preprint arXiv:2001.08499* (2020).

[26] Daniel Gehrig et al. “Video to events: Recycling video datasets for event cameras”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3586–3595.

[27] RPG Lab. *Ultimate SLAM: A combined framework for visual-inertial, event-based, and stereo SLAM*. https://github.com/uzh-rpg/rpg_ultimate_slam_open. Accessed: 2024-07-02. 2020.

[28] Etienne Perot et al. “Learning to detect objects with a 1 megapixel event camera”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 16639–16652.

[29] Automate.org. *Luca Thumbnail*. https://www.automate.org/images/ogImages/Luca-Thumbnail-1.png. 2021.

[30] Fuqiang Gu et al. *EventDrop: data augmentation for event-based learning*. 2021. arXiv: 2106.05836 [cs.LG].

[31] Henri Rebecq et al. “High Speed and High Dynamic Range Video with an Event Camera”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.6 (2021), pp. 1964–1980. DOI: 10.1109/TPAMI.2019.2963386.

[32] Guillermo Gallego et al. “Event-Based Vision: A Survey”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.1 (2022), pp. 154–180. DOI: 10.1109/TPAMI.2020.3008413.

[33] Weipeng Guan and Peng Lu. "Monocular Event Visual Inertial Odometry based on Event-corner using Sliding Windows Graph-based Optimization". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 2438–2445. DOI: 10.1109/IROS47612.2022.9981970.

[34] Yuhang Li et al. "Neuromorphic data augmentation for training spiking neural networks". In: *European Conference on Computer Vision*. Springer. 2022, pp. 631–649.

[35] Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. "AEGNN: Asynchronous Event-Based Graph Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 12371–12381.

[36] Yi–Fan Zuo et al. "DEVO: Depth-Event Camera Visual Odometry in Challenging Conditions". In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 2179–2185. DOI: 10.1109/ICRA46639.2022.9811805.

[37] Mathias Gehrig and Davide Scaramuzza. "Recurrent Vision Transformers for Object Detection With Event Cameras". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 13884–13893.

[38] Nikola Zubić et al. "From Chaos Comes Order: Ordering Event Representations for Object Recognition and Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 12846–12856.

[39] Nikola Zubić et al. "From Chaos Comes Order: Ordering Event Representations for Object Recognition and Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 12846–12856.

[40] Shuang Guo and Guillermo Gallego. "CMax-SLAM: Event-Based Rotational-Motion Bundle Adjustment and SLAM System Using Contrast Maximization". In: *IEEE Transactions on Robotics* 40 (2024), pp. 2442–2461. DOI: 10.1109/TRO.2024.3378443.

[41] Wikipedia. *Inertial Measurement Unit*. Accessed: 2024-05-22. 2024. URL: https://en.wikipedia.org/wiki/Inertial_measurement_unit.

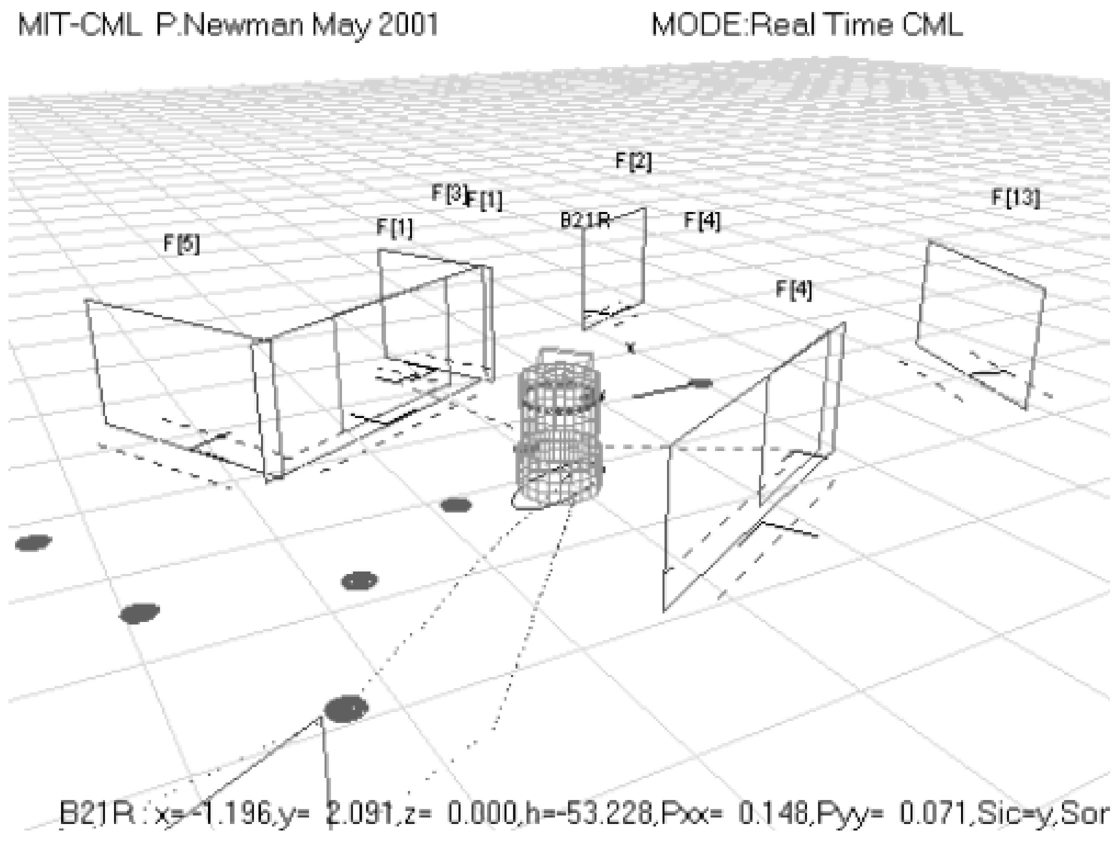
[42] Hu Zhang et al. *Automotive Object Detection via Learning Sparse Events by Spiking Neurons*. 2024. arXiv: 2307.12900 [cs.CV].

[43] Nikola Zubic, Mathias Gehrig, and Davide Scaramuzza. "State space models for event cameras". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 5819–5828.

[44] Prophesee. *Metavision Framework*. https://www.prophesee.ai/metavision-intelligence/. Accessed: 2024-03-06.

CHAPTER A

Advancements in SLAM: Harnessing the Power of Event Cameras



```
B21R : x=-1.196,y= 2.091,z= 0.000,h=-53.228,Pxx= 0.148,Pyy= 0.071,Sic=y,Sor
```

Figure A.1: During the exploration phase, the robot was operated manually, but the operator did not have direct visual contact. Instead, they navigated the robot by relying on a real-time map rendered from the robot's sensor data [4].

In the evolving landscape of robotic navigation and computer vision, Simultaneous Localization and Mapping (SLAM) stands out as a key technology, enabling autonomous systems to decipher and interact with their environments. SLAM faces the complex problem of finding appropriate representations for both the observation and the motion models, necessitating a sensorial system capable of capturing measurements of the relative location between landmarks and the vehicle itself. This chapter will explore the essence of SLAM, its distinctive features, the various types existing, and the integration of event cameras into SLAM systems, a development that promises to revolutionize how machines perceive motion and spatial complexity. Event cameras, with their unique ability to detect changes in light per pixel, enhance the capability of SLAM systems to operate in dynamic and visually challenging environments.

A.1 Fundamentals and Applications of SLAM

Simultaneous Localization and Mapping (SLAM) is a critical process in the field of robotics and computer vision (see Figure A.1). SLAM allows a robot or an autonomous vehicle to map an unknown environment while simultaneously keeping track of its position within that map. This is essential in contexts where GPS is not usable, such as inside buildings or in underwater and underground environments.

The idea of SLAM was born in the 1980s when researchers began to seriously explore the problem of simultaneous localization and mapping. The term "SLAM" was introducted in the 1990s when the field started to receive significant attention from the academic community. One of the early influential works was by Hugh Durrant-Whyte and John J. Leonard in 1991 [1], which explored landmark-based localization for mobile robots, as well as a method in which the robot used predefined or easily recognizable points in the environment to determine its own location. These landmarks could be natural features (e.g., rocks, trees, terrain configurations) or artificial ones (e.g., barcodes, QR codes, specific reflectors), installed in the environment. This approach was particularly useful in contexts where GPS was unavailable or unreliable. These pioneering studies laid the groundwork for subsequent research and developments in the field.

SLAM has found applications in a wide range of areas:

- *Robotics*: SLAM is fundamental for the autonomous navigation of robots in unknown environments, such as home robots, drones, and underwater vehicles.
- *Autonomous Vehicles*: For driverless cars, SLAM helps navigate complex streets and urban environments without GPS support.

- *Augmented and Virtual Reality*: In AR and VR, SLAM allows the device to understand and interact with the user's real environment, enhancing the immersive experience.
- *Space and Underwater Exploration*: SLAM is essential in environments where extreme conditions or isolation make other forms of navigation impractical.

For the technology, various tools and sensors are used to collect data and process it effectively. Here is an overview of the most common tools used in SLAM:

- *Lidar (Light Detection and Ranging):* Uses laser pulses to measure distances and create detailed 3D maps.
- *Sonar (Sound Navigation and Ranging):* Uses sound waves, useful in aquatic environments or where optical visibility is limited.
- *Radar:* Uses radio waves to detect distances and speeds, functioning well even in adverse weather conditions.
- *Stereo Cameras:* Two or more cameras for stereo triangulation and depth calculation.
- *RGB-D Cameras:* Provide color images and depth information through infrared sensors.
- *IMU (Inertial Measurement Unit):* A combination of an accelerometer, gyroscope, and sometimes a magnetometer, to track movement and orientation.

Software and Algorithms:

- *ROS (Robot Operating System):* A framework for robotic development, with libraries for SLAM.
- *Gmapping:* A ROS package that implements SLAM with an extended Kalman filter for 2D mapping.
- *RTAB-Map (Real-Time Appearance-Based Mapping):* Optimized for real-time executions on robots and autonomous vehicles.
- *ORB-SLAM:* Uses ORB descriptors for efficient environment detection and mapping.
- *Cartographer:* Supports Lidar and IMU for 2D and 3D mapping, developed by Google.

Despite significant advancements, SLAM technology still faces different challenges, such as the difficulty in managing large maps, the need for robustness in dynamic environments, and the effort to lower energy consumption.

A.2 SLAM Data Collection

The collection of data for SLAM (Simultaneous Localization And Mapping) is a fundamental component to ensure that the system can accurately localize itself and map the surrounding environment. Two common approaches in SLAM data collection are Visual Odometry and Visual-Inertial Odometry, each with its own methods and advantages.

A.2.1 Localization, Ego-motion Estimation

In SLAM data collection, localization and ego-motion estimation are critical aspects. Localization allows the device to determine its position within an environment, while ego-motion estimation tracks how this position changes over time. These processes are essential for building accurate maps and navigating effectively.

A.2.2 Visual Odometry (VO)

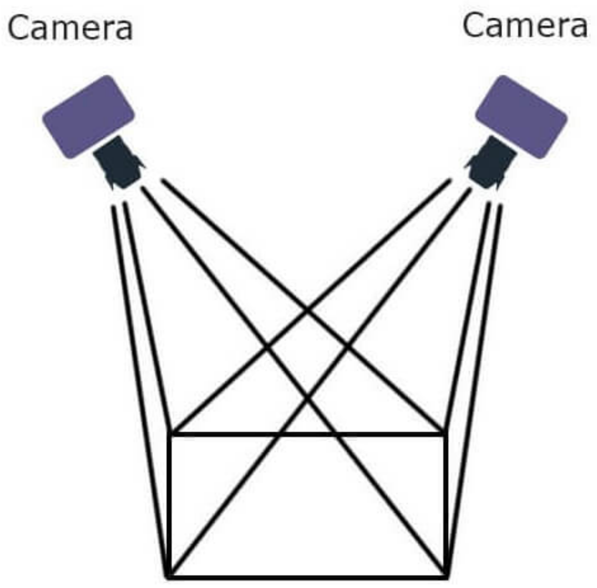


Figure A.2: A stereo camera with its two visible lenses projects its view to create a depth map of the surrounding environment. The converging lines of sight from the two lenses illustrate how corresponding points in the images captured by the two cameras are used to calculate disparity, which is essential for determining the depth of objects in the environment. This setup enables obtaining an accurate and detailed three-dimensional representation of the observed scene.

Visual Odometry (VO) uses visual sensors to estimate the device's movement by analyzing the images captured over time. Here are the main configurations used:

- *Monocular Cameras*: A single camera captures two-dimensional images. By analyzing these images, feature points can be detected and tracked between consecutive frames to estimate movement. However, a monocular camera cannot directly perceive depth, requiring advanced image processing techniques to estimate distances.
- *Stereo Cameras*: A system with two or more cameras captures images simultaneously from slightly different perspectives. This setup allows the depth perception through triangulation, improving the accuracy of movement estimation (Figure: A.2).

VO is advantageous because it works with relatively simple hardware and provides detailed information about the surrounding environment. However, it is sensitive to the variation of lighting conditions and to the lack of texture of the scenes, and it can accumulate errors (drift) over time without other forms of correction.

A.2.3 Visual-Inertial Odometry (VIO)

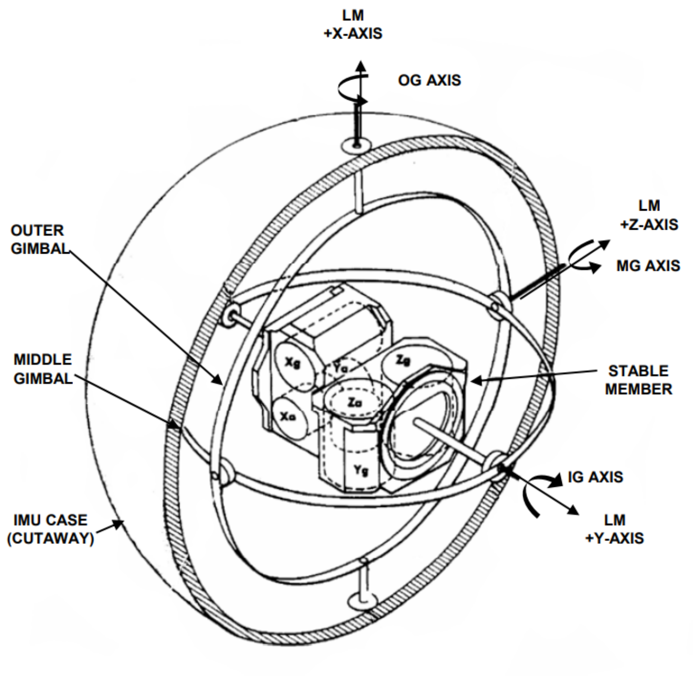


Figure A.3: Inertial Reference Integrating Gyros (IRIGs,Xg,Yg,Zg) sense attitude changes, and Pulse Integrating Pendulous Accelerometers (PIPAs,Xa,Ya,Za) sense velocity changes [41].

Visual-Inertial Odometry (VIO) integrates visual sensors with inertial sensors (IMU - Inertial Measurement Unit, which includes accelerometers and gyroscopes) to enhance the robustness and accuracy of movement estimates. The IMU (Figure: A.3) provides high-frequency data regarding changes in acceleration and rotation, which are integrated with visual information for more precise movement estimation.

VIO offers several advantages over VO alone:

- *Increased Robustness and Accuracy*: The combination of visual and inertial data makes VIO less susceptible to errors caused by variation of lighting conditions or by rapid movements.
- *Reduction of Error Accumulation (Drift)*: Integrating IMU data helps correct accumulated errors over time, improving movement estimation accuracy.
- *Better Handling of Challenging Environments*: VIO is more effective in environments where visual data alone may be unreliable, such as in low-light conditions or scenes with few textures.

However, VIO requires the integration and calibration of multiple sensor types, increasing hardware and software complexity.

So, while Visual Odometry uses only visual sensors to estimate movement and build maps, Visual-Inertial Odometry integrates these sensors with inertial units to improve system accuracy and robustness. This combination makes VIO a preferred choice for applications requiring high reliability and accuracy in SLAM data collection.

A.3 Extract and Trace Information from Images

After having thoroughly examined the concept of SLAM and its two main types, we now explore how the processing of collected data has evolved over time.

Node creation in the SLAM graph is based on identifying common points between image sequences, allowing the tracking of camera movement within the environment. But how exactly does this tracking occur?

With the advent of deep learning, computer vision has seen significant improvements. Detecting features, edges, and corners has turned into routine for these advanced algorithms. However,

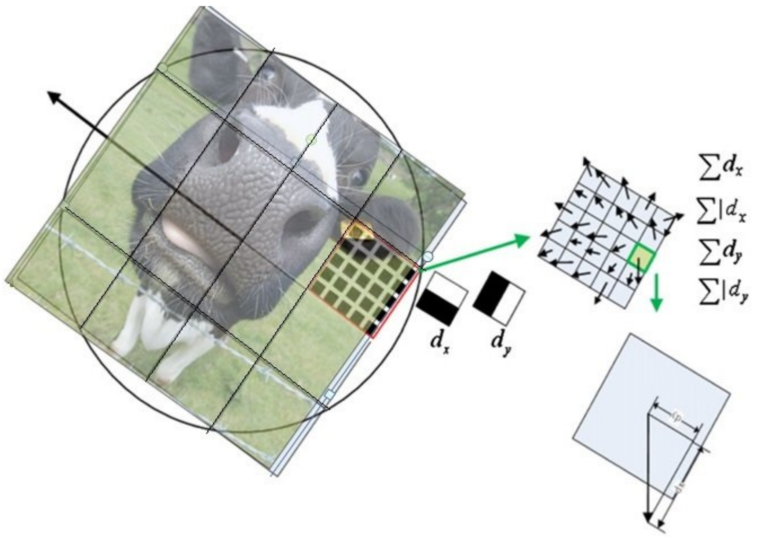


Figure A.4: This image illustrates gradient-based algorithms that detect edges by calculating changes in intensity across pixels. These methods are essential for identifying key features in images, facilitating object recognition and image segmentation.

the use of neural networks in this tracking phase entails a high computational expense. Consequently, it is essential to seek solutions that effectively balance accuracy and computational resource management.

Before the era of deep learning, tracking was dominated by computer vision techniques that employed algorithms such as SIFT (Scale-Invariant Feature Transform) [5], SURF (Speeded Up Robust Features) [6], ORB (Oriented FAST and Rotated BRIEF) [10], and FAST (Features from Accelerated Segment Test) [8]. These algorithms, based on gradients and transformations (Figure:A.4), detected features, edges, and corners, extracting crucial information from various frames in a mathematically sophisticated yet computationally manageable way. These algorithms represented an ideal compromise for tracking.

A.4 Types of SLAM

After a general overview of SLAM, it's essential to explore the topic in a more detailed view. There are various types of SLAM, which can be mainly divided into two categories: filter-based SLAM and optimization-based SLAM, also known as graph-based SLAM. Each of these categories includes different variants and specific techniques, adapting to various contexts and application needs. In the following sections, we will discuss these two types to better understand their differences and try to determine their advantages and disadvantages.

A.4.1 Filter-Based SLAM

Filter-based SLAM uses probabilistic algorithms for the continuous estimation of the robot's state and the map of the environment. This method is closely related to the use of Bayesian filters, such as the Extended Kalman Filter (EKF) and FastSLAM. Both approaches aim to solve the problem of simultaneously determining the robot's position in the environment and mapping the location of fixed reference points (landmarks).

- *Extended Kalman Filter (EKF) [2]:*

  The Extended Kalman Filter is a predictive algorithm that estimates the state of a dynamic system in the presence of uncertainties. Specifically, the EKF extends the Kalman filter to handle nonlinear models through linearization, allowing for the update of the state estimate and its uncertainty (covariance). In this context, covariance is a measure that describes how much the variables (robot's position and landmarks) vary and how these variations are correlated with each other. In SLAM, the EKF updates a global covariance matrix that accounts for all uncertainties associated with both the robot's position and the landmarks, making the process computationally intensive, especially with a large number of landmarks.

- *FastSLAM [3]:*

  FastSLAM, on the other hand, uses a combination of a particle filter and Kalman filters to manage separately the estimation of the robot's paths and the position of landmarks. A "particle" in FastSLAM represents a possible "history" of the robot's path, each with its associated estimate for the position of the landmarks. This approach divides the complex problem into many smaller, more manageable subproblems, each of which is simpler to solve thanks to the "conditional independence" of the landmark positions relative to the known path.

Thanks to its innovative structure, FastSLAM is significantly more scalable than EKF SLAM. While EKF requires computational time that grows quadratically with the number of landmarks, FastSLAM grows only logarithmically thanks to the use of binary trees to manage landmark updates. This makes FastSLAM particularly suitable for environments with a large number of reference points, where EKF SLAM might not be feasible. In the table A.1, you can find a summary of the fundamental characteristics of both methodologies.

Table A.1: Comparison between EKF SLAM and FastSLAM

Feature	EKF SLAM	FastSLAM
Computational Complexity	Grows quadratically with the number of landmarks.	Grows logarithmically with the number of landmarks.
Estimation Methodology	Uses a common covariance matrix for both the robot and landmarks.	Separates the robot's path estimation from the landmarks using Kalman filters for the landmarks and a particle filter for the robot's path.
Scalability	Limited to a few hundred landmarks.	Effective up to 50,000 landmarks.
Data Association Management	Association errors can lead to rapid failures.	More robust against association errors, manages multiple possible associations per particle.
Resource Usage	Requires complex updates to the covariance matrix.	Uses an efficient tree-based structure to reduce complexity.

A.4.2 Optimization-Based SLAM (Graph-Based SLAM)

Optimization-based SLAM, also known as graph-based SLAM, is an alternative to filter-based SLAM, focusing on a global problem-solving approach rather than real-time and continuous estimation. This method constructs a graph where nodes represent the robot's positions at different times, and edges represent relative measurements between positions, such as distances and angles measured by sensors.

- *Graph Construction*

  Graph construction begins with the creation of a node for each recorded position of the robot over time. Each node is then connected to others through edges that represent sensory observations or movements of the robot. For example, if the robot moves from point A to point B, this transition will be represented by an edge between the two corresponding nodes.

- *Graph Optimization*

  Once the graph is built, the goal is to optimize the arrangement of nodes so that the measurements (edges) are as consistent as possible with the sensory data. This is achieved through nonlinear optimization algorithms, such as the Gauss-Newton or Levenberg-Marquardt methods, which minimize the discrepancy between the observed measurements and those predicted by the graph model.

This type of SLAM is particularly effective in handling large amounts of data and uncertainties accumulated over time, and produces a global optimization that can correct errors that have accumulated in previous estimates. Moreover, graph-based SLAM can reconsider and reprocess past information if new measurements indicate that previous estimates could be improved.

In the table A.2 a comparison between the features of graph-based SLAM and filter-based SLAM is summarized.

Table A.2: Comparison between Filter-Based SLAM and Graph-Based SLAM

Characteristic	**Filter-Based SLAM**	**Graph-Based SLAM**
Approach	Real-time estimation, continuous state update.	Global resolution, optimization based on overall model coherence.
Computational Complexity	More suitable for real-time applications with fewer landmarks.	Requires more computational resources, but better manages large data sets.
Robustness	Sensitive to large initial estimation errors and noisy measurements.	More robust against error accumulation and capable of retrospective corrections.
Scalability	Limited by growing computational needs with the number of landmarks.	Highly scalable, suitable for applications requiring extensive mapping.
Adaptability	Suitable for less complex environments or with less need for retrospective precision.	Excels in dynamic and complex environments where data can be frequently updated.

These two main categories of SLAM offer different approaches to solving the problem of simultaneous localization and mapping, each with its strengths and limitations that must be considered based on the specific context of use.

Implementation of Graph-Based SLAM

Now, we will describe the implementation process of Graph-Based SLAM, highlighting the importance of each phase in the context of a dynamic robotic environment.

- **Step 1: Data Collection**

  In Graph-Based SLAM, data collection is crucial and relies on the use of a variety of sensors. Here are the types of data collected:

  - *Sensors Used:* Lidar, cameras, ultrasonic sensors, etc.
  - *Data Collected:* Distances, images, angles.

  The combination of these data allows the SLAM system to have a detailed understanding of the environment and how the robot moves within it. This phase is fundamental because it provides the base of information on which all subsequent phases of the SLAM process are built.

- **Step 2: Graph Creation**

  In the graph of Graph-Based SLAM, each node represents an estimate of position, which can be a specific position of the robot at a given time or the position of a landmark in the environment. Accuracy in defining the nodes is crucial, as these form the basic structure on which the entire mapping and localization model is built.

  - *Robot Positions:* Nodes related to the robot are generally added every time the robot reaches a new significant position or after traveling a predefined distance.
  - *Landmarks:* Landmark nodes are added when a specific object or reference point is identified and deemed useful for the navigation and mapping process.

  Creating the graph is a critical process that involves not only the collection and integration of sensory data but also the intelligent interpretation of these data to form a coherent and useful representation of the environment and the robot's trajectory. Efficient management of the graph is essential to ensure that the SLAM system is scalable and manageable as the environment becomes larger and more complex.

- **Step 3: Adding Edges**

  The edges in the Graph-Based SLAM graph represent metric relationships, or geometric measurements between nodes that may include distances and relative angles. The importance of the edges is crucial for maintaining the structural integrity of the graph, ensuring that the relationships between the positions are consistent and realistic compared to the physical measurements of the environment.

  – *Metric Relationships:* Geometric measurements between nodes.

  – *Importance of Edges:* Maintaining the structural integrity of the graph.

- **Step 4: Graph Optimization**

  Optimizing the graph is fundamental to minimizing the global error in estimating positions and mapping. Optimization algorithms such as Gauss-Newton and Levenberg-Marquardt are employed, specializing in managing non-linear optimization problems, to refine position estimates and improve the precision of the resulting map.

  – *Optimization Algorithms:* Gauss-Newton, Levenberg-Marquardt.

  – *Non-Linear Optimization:* Minimizing the global error.

- **Step 5: Iteration and Refinement**

  The SLAM process is iterative and requires continuous cycles of refinement to improve the map and the robot's trajectory. During these iterations, uncertainties are also managed through stochastic models, enhancing the robustness of the navigation system.

  – *Iterations:* Iterative process of refinement.

  – *Management of Uncertainties:* Stochastic modeling of noise and uncertainties.

- **Step 6: Finalization of the Map**

  This final step realizes the work of SLAM in a detailed map and a reliable trajectory of the robot. These results are essential for practical applications such as autonomous navigation and route planning, demonstrating the effectiveness of the SLAM system in real environments.

  – *Final Results:* Detailed map and reliable trajectory of the robot.

  – *Practical Applications:* Autonomous navigation, route planning.

The process of implementing Graph-Based SLAM, described through the various steps above, is fundamental to understanding the basics of how this technology works and its application in dynamic robotic contexts. It is important to emphasize that, although these steps provide a general framework, the field of graph-based SLAM is continuously evolving, with the development of new types of graphs and optimization techniques.

A.5 The Importance of Integrating the Event Camera in SLAM

As discussed in previous chapters, the event camera offers a series of unique advantages that can significantly improve the performance of SLAM. This technology is distinguished by its extremely low latency, high dynamic range, and motion blur reduction, capabilities that allow it to capture more information than a standard camera in complex environmental conditions. It is also important to highlight the energy efficiency and reduced memory requirements: unlike lidar and RGB cameras, which require significant amounts of energy, the event camera offers a more sustainable and efficient solution.

It should be noted that the use of the event camera in the world of SLAM is relatively recent, and therefore, the optimization of its use in this context is still ongoing.

Scientific research on SLAM using the event camera is continually evolving and is currently focused on refining tracking. Since the event camera operates asynchronously and not by frames, research efforts are aimed at improving feature detection. Therefore, we still are in the initial phase of tracking, and further refinement is likely needed before proceeding to the next phase of semantics.

Various technologies that will contribute to the refinement of tracking are being explored. The first [21] [40] are based on the acquisition of events and their periodic transformation into frames, in order to use the detection algorithms mentioned in section A.3. This approach allows us to exploit all the advantages of the event camera, but involves the loss of some computational benefits, as it is still necessary to process matrices. The next approach [33] [36] being tested involves implementing algorithms that can track using events only.

In conclusion, we can assert that the introduction of the event camera in the field of SLAM is likely to make significant contributions over the long term. However, it is crucial not to overlook the limitations of this technology under conditions of stasis, where there is no variation in light intensity, and the event camera fails to capture data, unlike the RGB camera.

From this perspective, there is a clear opportunity for optimal complementarity between the two technologies. It would therefore be advisable to develop a system that favors the use of the event camera as the primary sensing tool, but that can activate the RGB camera in situations of environmental stasis. This approach would leverage the strengths of both technologies, ensuring that the most accurate and complete information is available under all circumstances.

CHAPTER B

Advances in Object Detection with Event Cameras

In the rapidly evolving fields of computer vision and artificial intelligence, object detection has emerged as a crucial technology. It enables systems to recognize and classify objects within an environment, paving the way for numerous applications ranging from autonomous driving to security surveillance. This chapter explores the essence of object detection, highlighting its fundamental principles, diverse methodologies, and the integration of advanced technologies such as deep learning and sensor fusion that promise to enhance the accuracy and efficiency of detection systems.

B.1 Overview of Deep Learning

Before discussing object detection, it is appropriate to mention deep learning, as object detection relies entirely on it. Deep learning is a subcategory of machine learning that uses artificial neural networks with many layers (hence the term "deep") to model and analyze complex data. These deep neural networks are inspired by the structure and functioning of the human brain, with artificial neurons (nodes) organized into layers that process information through weighted connections.

The main features of *deep learning* include:

– *Layering*: Deep neural networks are composed of many layers of neurons, each processing information at an increasingly abstract and complex level.

- *Supervised and Unsupervised Learning*: *Deep learning* can be applied in both supervised (with labels) and unsupervised (without labels) modes to learn data features.
- *Applications*: It is used in various applications such as image recognition, speech recognition, machine translation, autonomous driving, and much more.
- *Training Algorithms*: It utilizes advanced algorithms like gradient descent and optimization techniques to train neural networks, updating the weights of the connections between neurons to minimize prediction errors.
- *Big Data and Computational Power*: It requires large amounts of data and significant computational power, often provided by graphics processing units (GPUs) or TPUs, to be effective.

Deep learning has revolutionized many fields, offering superior performance in many areas compared to traditional *machine learning* methods.

B.2 Fundamentals and Applications of Object Detection

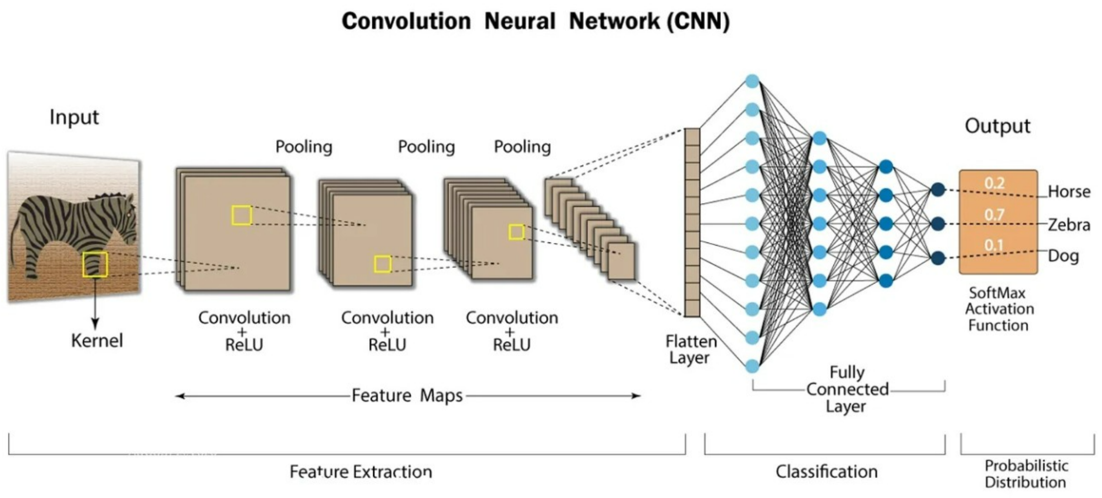


Figure B.1: The photo depicts an image of a zebra processed by a Convolutional Neural Network (CNN) in three main stages. On the left, the input image is processed by convolutional layers that extract relevant features. In the center, pooling layers reduce the dimensions while retaining essential information. On the right, fully connected layers combine the extracted features to produce the final classification.

Object detection is a fundamental process in computer vision that involves identifying and locating objects within an image or a video. It not only determines the presence of

objects but also their positions, typically through bounding boxes. This capability is vital in scenarios where understanding and interacting with the environment is essential.
The concept of object detection can be traced back to the early days of computer vision, but it has significantly evolved with the advent of machine learning and deep learning. Initial methods relied on handcrafted features and simple classifiers, but modern approaches leverage Convolutional Neural Networks (CNNs)(see Figure B.1) and advanced algorithms to achieve remarkable performance. A Convolutional Neural Network (CNN) is a type of deep neural network particularly effective for processing grid-structured data, such as images.
Object detection has a wide range of applications, including:

- *Autonomous Vehicles*: Object detection is crucial for identifying pedestrians, vehicles, traffic signs, and other obstacles, ensuring safe navigation.
- *Surveillance and Security*: Automated systems can detect intruders, recognize faces, and monitor activities in real-time.
- *Inventory Management and Retail*: In retail, object detection helps track inventory, analyze shopper behavior, and enhance customer experience.
- *Healthcare*: Used in medical imaging to detect anomalies, such as tumors, in X-rays and MRIs.
- *Robotics*: Assists robots in identifying and interacting with objects, crucial for tasks like picking, placing, and assembling.

B.3 Technologies and Tools in Object Detection

The field of object detection employs a variety of sensors and computational tools to achieve high accuracy and robustness. Here are some of the most prominent:

- *Deep Learning Frameworks*: Tools such as TensorFlow, PyTorch, and Keras facilitate the development of sophisticated neural networks for object detection.
- *Pre-trained Models*: Models like YOLO (You Only Look Once) [15], SSD (Single Shot MultiBox Detector)[14], and Faster R-CNN (Region-based Convolutional Neural Networks)[16] provide robust solutions for real-time object detection.

– *Edge Devices*: Hardware accelerators such as NVIDIA Jetson and Intel Movidius support efficient inference on edge devices, making real-time detection possible even in resource-constrained environments.

– *Sensor Fusion*: Combines data from multiple sensors, such as cameras, Lidar, and Radar, to improve the accuracy and reliability of detection.

B.4 Object Detection and Data Annotation

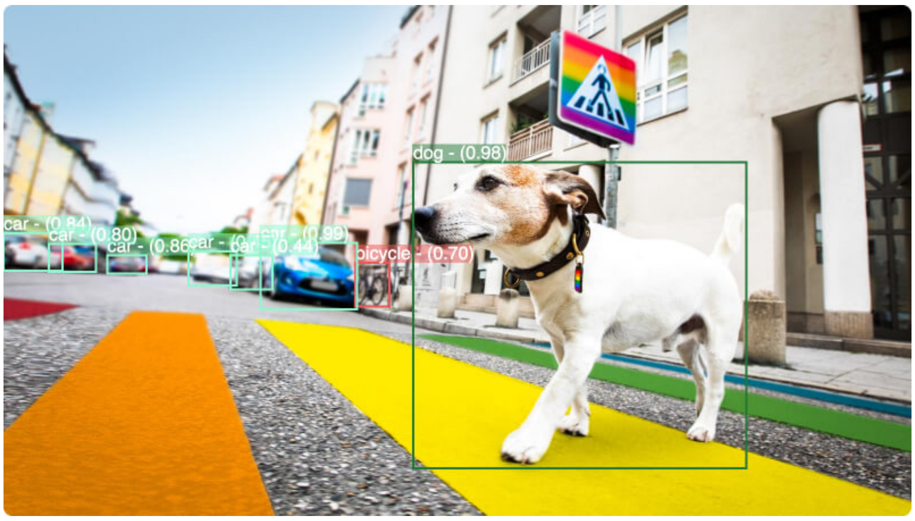


Figure B.2: The image shows the use of bounding boxes with a dog walking, highlighted by a bounding box labeled "dog" with a confidence score of 0.98. Other detected objects include a bicycle (0.70) and several cars with varying confidence scores. The colorful street and buildings in the background add context to the scene.

Data collection is a crucial component to ensure that object detection systems can operate with high precision and reliability. The first step is to use a camera to capture photos or video frames. Subsequently, to train a neural network to recognize specific targets, it is extremely important to select and classify each pixel as a certain instance. This process, known as data annotation, requires meticulous attention to detail.

Experts who perform this task are known as *data annotators*. Their job is to draw *bounding boxes* around objects of interest in images. A bounding box is a rectangle that encloses the object and defines its boundaries. Precision in annotation is crucial: if an annotator incorrectly includes or excludes pixels, the neural network might learn to recognize objects inaccurately. For instance, if we are training a network to recognize people and some people are correctly annotated while others are not, the network might learn to recognize people only in certain cases and consider them as background in others. This error is unacceptable, especially when the technology is used for critical applications like security, autonomous driving, or healthcare.

There are various techniques for detecting and annotating objects:

– *Bounding Box Annotation*: Drawing rectangles around objects of interest in images, defining their boundaries(Figure: B.2.
– *Mask Annotation*: Drawing more precise contours around objects, allowing for more detailed segmentation, particularly useful for objects with irregular shapes.
– *Point Annotation*: Identifying and annotating specific key points of objects, such as facial features.
– *Polygon Annotation*: Creating polygons that precisely follow the contours of the object, allowing for accurate segmentation of complex objects.
– *3D Cuboid Annotation*: Used to annotate objects in a three-dimensional context, providing information on depth as well as size and position.
– *Polyline Annotation*: Used to trace lines and paths, useful in applications like road and route recognition in maps.

B.5 Challenges and Future Directions

Despite the advances, object detection technology still faces several challenges:

– *Occlusions and Variability*: Detecting objects that are partially obscured or come in various shapes and sizes remains difficult.
– *Real-Time Performance*: Achieving high-speed detection without compromising accuracy is crucial for applications like autonomous driving.
– *Adverse Conditions*: Performance can decline in low light, bad weather, or cluttered environments.
– *Energy Consumption*: High computational demands often lead to increased energy consumption, which is a concern for battery-powered devices.

B.6 The Importance of Integrating the Event Camera in Object Detection

In the context of object detection, the use of event cameras represents a significant innovation compared to traditional RGB cameras. Event cameras offer a range of advantages that can greatly enhance object detection capabilities, especially in dynamic conditions and environments with variable lighting. Therefore, efforts are made to eliminate or at least mitigate the problems listed in the previous section. However, as event cameras are a much more recent technology compared to standard RGB cameras, there are some challenges to overcome.

B.6.1 Challenges in Using Event Cameras

Despite their numerous advantages, the adoption of event cameras in the field of object detection presents some challenges:

- *Processing Algorithms:* Object detection algorithms need to be adapted to work with the unique data format produced by event cameras. Neural networks and other machine learning models must be specifically trained to interpret and utilize events instead of traditional RGB frames.
- *Temporal Synchronization:* The asynchronous nature of events requires advanced techniques to synchronize and aggregate temporal data in a way that is useful for object detection. This can complicate integration with other detection technologies.
- *Data Interpretation:* Event data is often more complex to interpret than RGB image pixels. The spatial and temporal representation of events requires new methodologies for visualization and analysis.

B.6.2 Technological Solutions

Recent studies have shown that various approaches have been explored for event-based object detection. Here we summarize the main technologies and methodologies discussed in the recent literature:

1. *Dynamic Graph Neural Networks (GNNs)* [35]: Recent studies have demonstrated that dynamic GNNs can theoretically achieve low-latency inference for event-based

object detection. However, for practical scenarios, they require specialized hardware or improvements in detection performance.

2. *Sparse Neural Networks* [42]: Similar to GNNs, sparse neural networks can achieve low latency, but they also need specialized hardware or improvements in detection performance.

3. *Dense Neural Network Design* [28]: Some work focuses on conventional dense neural network designs, showing impressive performance in event-based object detection, particularly when utilizing temporal recurrence in their architectures. However, the processing latency of these approaches remains over 40 milliseconds, limiting the ability to fully exploit the low latency of event cameras.

4. *Convolutional LSTM (Conv-LSTM)* [13]: Previous work extensively uses Conv-LSTM cells in their feature extraction stage or relies on heavy architectures like VGG, which entails a significant computational cost.

5. *Temporal Recurrence* [37]: It has been found that simple LSTM cells, operating separately on each feature, can replace Conv-LSTM cells, drastically reducing the number of parameters and latency while improving overall performance.

These technologies and methodologies collectively enable an excellent compromise between detection performance and inference times, significantly reducing parameter counts and inference times compared to previous approaches.

CHAPTER 8

Ringraziamenti

Desidero esprimere la mia più profonda gratitudine al Professor Marco Cococcioni per il supporto e la guida forniti durante il periodo della mia tesi magistrale.

Ho scelto lui come relatore della mia tesi per una ragione molto particolare. Quando ho iniziato il mio percorso universitario alla triennale, non avevo la più pallida idea di come affrontare gli esami, e la situazione era resa ancora più difficile dalla necessità di salvare la borsa di studio, senza la quale non avrei potuto proseguire gli studi. Il 29 luglio del 2016 rappresentava l'ultima mia opportunità per mantenere la borsa di studio, ma l'esame stava andando molto male e la collega esaminatrice era incline a bocciarmi.

Il Professor Cococcioni, conoscendo la mia situazione, ha deciso di darmi una possibilità che raramente viene concessa agli studenti. Ha convinto la collega a farmi accomodare tra i posti a sedere, permettendomi di riflettere con calma sulla domanda, mentre gli altri studenti continuavano con gli orali. Quella semplice decisione ha letteralmente cambiato le sorti della mia vita.

Per me, è stato un grandissimo gesto di umanità che porterò sempre dentro di me.

Grazie di cuore, Professor Cococcioni.

Vorrei esprimere esprimere la mia sincera gratitudine al Dottor Niccolò Camarlinghi per il suo supporto fondamentale nei primi tre mesi della mia tesi e per il suo continuo interesse nei miei progressi anche nei successivi tre mesi.

Il Dottor Camarlinghi è stato un punto di riferimento essenziale, riuscendo a tranquillizzarmi nei momenti di maggiore confusione. Sapere di poter contare su di lui mi ha dato

un grande sollievo. Anche quando aveva tutte le ragioni del mondo per ignorarmi e non prestarmi supporto a causa di situazioni personali molto più importanti, mi ha comunque assistito con grande forza.

Mi ha davvero insegnato tanto sul valore della parola e della pazienza. La sua presenza e il suo aiuto costante hanno fatto una grande differenza nel mio percorso.

Grazie di cuore, Dottor Camarlinghi.

Eccomi, ho finito il mio percorso. Tutto è iniziato a settembre del 2015. È giusto che sappiate perché ho iniziato. Non mi è mai piaciuto impegnarmi troppo per capire quale fosse la mia strada. Mi è sempre piaciuto buttarmi nell'onda senza pensarci troppo. D'altronde questo percorso ne è la dimostrazione. A scuola non studiavo, i professori volevano bocciarmi, le uniche materie nelle quali andavo bene senza studiare erano matematica e informatica. Quindi le strade erano chiare. Matematica però all'epoca pensavo potesse portarmi a fare solo il professore, informatica era interessante, ma ingegneria informatica suonava proprio bene. Mi sono buttato nella prima onda. Ma potevo mai sapere che andando avanti c'erano gli tsunami?

Diciamolo francamente, la prima volta che feci il test per accedere al Politecnico di Torino non posso dimenticare i dati statistici in base al mio punteggio. Solamente circa il 3/4% di quelli riuscivano a laurearsi. Arrivo all'università di Pisa e gente diplomata con il massimo dei voti comincia a ritirarsi perché non ce la fa. In tutto questo io non stavo davvero capendo nulla, molti ragazzi degli anni successivi mi dicevano di scappare. L'unica domanda che mi ponevo era: perché se altri prima di me ci sono riusciti io non dovrei?

Sapete cosa mi ha salvato? La mia superficialità nel prendere la vita con leggerezza e senza pensare troppo alle conseguenze, la mia competitività e la rabbia che nutrivo per il sistema. Non potevo farmi sottomettere da un sistema che permette a una persona con un semplice voto di far sentire un'altra persona elogiata o umiliata a sua completa discrezione. Ho scoperto di avere una determinazione incredibile, non mi interessava fare brutta figura di fronte al professore se non avevo capito bene il programma. Per rispetto del mio tempo, io ci andavo e se mi bocciava ritornavo. Gli esami sono 21 alla triennale e 12 alla magistrale, 33 in totale (credo che non sia un caso che corrisponda all'età di Cristo), il mio libretto elettronico conta all'incirca 110 iscrizioni agli esami. Ammetto di aver pianto davanti a delle materie e che alcune mi hanno fatto pensare seriamente che non fosse la mia strada. Insomma, non so quante persone avrebbero scommesso 1 euro sul me di 18 anni. D'altronde è comprensibile, nemmeno io lo avrei fatto.

C'è chi pensa che determinate strade siano fatte solo per chi ha talento ed è portato a fare quella cosa, io non ci credo. Nulla è impossibile, basta crederci e perseverare. Qui riporto 2 frasi che sono presenti come didascalia nel mio profilo Instagram: "È difficile ma realizzabile o realizzabile ma troppo difficile? Dipende tutto ed esclusivamente da noi, da come ci poniamo davanti alla vita". Ecco, questa storia è racchiusa nelle parole di quella chiavetta "cu l'avissi rittu".

È stato un percorso durato quasi 9 anni, sono tanti, ma sono stati così pieni di emozioni di quelle che ti fanno sentire vivo. Ho conosciuto un botto di gente, ho vissuto in centinaia di personalità diverse e mi sono divertito un sacco. Non mi pento di nulla. Mi verrebbe da ringraziare me stesso per quello che ho raggiunto, ma nel mio caso, non posso prendermi questo merito. Questo l'ho capito in Lussemburgo, dove sono stato in un ambiente nuovo, con una lingua nuova che pur riuscendo a pronunciare delle parole per farmi capire, non riuscivo a trasmettere la mia persona, non riuscivo ad adattarmi come è mio solito fare. È stato uno dei periodi meno produttivi della mia vita. Mi sono reso conto che i miei risultati dipendono dalle persone che mi circondano. Se sto bene socialmente riesco a pensare e costruire, altrimenti no. È vero che il seme proviene da noi, se non ci mettiamo del nostro non facciamo niente, ma siamo persone e abbiamo bisogno di stimoli per andare avanti, e gli stimoli vengono con la condivisione della propria vita con altre persone. Ecco perché devo ringraziare tutti i presenti per gli stimoli che mi avete dato e soprattutto per essere qui.

Inizierei partendo proprio dall'inizio di questa magistrale. Una volta iniziata, pensavo fosse un continuo della triennale, ma non è stato così. Un capitolo nuovo si è aperto: la Duccios. La Duccios non è altro che una casa, brutta, ma che io mi ricorderò per sempre come qualcosa di meraviglioso. I protagonisti di questa casa siamo stati io, Peppino, Rosalia, Nichi e Chicca. Non ho mai stretto così tanto con un gruppo di persone. Ognuno di noi ha caratteristiche complementari all'altro. Siamo riusciti a creare qualcosa che si avvicinava di più a una famiglia che a una semplice amicizia. Per me siete tuttora i miei coinquilini anche se ognuno sta in case diverse già da 2 anni. Devo ringraziarvi tutti.

Peppino, siamo la stessa persona con gli stessi pensieri. È sempre un piacere parlare con te, sei quel tipo di amico sul quale so che potrei contare sempre. Rosalia, porti l'armonia di una mamma in casa. Nichi, mi manca vedere la fermentazione dei tuoi piatti in frigo, tvb. Chicca, che ti devo dire a te? Sei una bimba speciale. Grazie Duccio.

Subito dopo è accaduto il fattaccio. Tempo di arrivare a Praticelli, una ragazza si è appropriata del mio cuoricino, Ana. Iniziato tutto con un "fammi compagnia che devo fare shopping" trasformato in aperitivo e adesso stiamo insieme da quasi 2 anni. Il supporto che riesci a darmi tu è enorme, senza mai chiedermi nulla in cambio. Ma più del supporto, so quanto è difficile sopportare i miei capricci e starmi dietro sui miei progetti di vita futuristici. In questo periodo di tesi come non mai mi sei stata vicino e mi hai saputo consolare nei momenti più duri. Grazie per tutto. QB.

Devo poi ringraziare i miei colleghi Matteo, Peppe e Mattia. Colleghi ma anche ottimi amici. Se sono qui è anche merito vostro. Mi è piaciuto un sacco condividere progetti insieme, siete fortissimi. Ringrazio ovviamente anche Francesco Cartier, Leonardo Moccia, anche loro colleghi con i quali abbiamo visto le pene del diavolo alla triennale. Grazie. Grazie a Kliton, ex coinquilino e a quanto pare la leggenda narra che è anche lui iscritto al corso di ingegneria informatica, sicuramente un problema secondario dato la montagna di soldi che si fa già con il lavoro che andrebbe a fare dopo la laurea.

Devo ringraziare la famigerata Silvia Mazzotta, ho conosciuto praticamente prima lei che Pisa. Abbiamo condiviso così tante cose insieme. Quante risate.

Non dimentichiamoci di Paolo Carioti, ci vediamo poco, ma sai che sei nel mio cuoricino e che aspetto un cocktail offerto?

Non può mancare ovviamente il ringraziamento a mio fratello Mattia. Pensaci noi da Vittoria, insieme siamo arrivati a Pisa. Certo, capitava spesso che ci vedevamo per le vacanze di Natale, prendevamo il volo per Pisa e poi la volta dopo era all'aeroporto per tornare a Vittoria per Pasqua. Ma questo ci caratterizza, ci vogliamo bene lo stesso. Poi c'è Gabriellina, nonché nonna Ella, rinominata così dal sottoscritto per la buona cucina tradizionale che riesce a portare nelle nostre tavole. Quando penso a te a Pisa mi viene in mente il periodo covid che facevamo le mangiate illegali. Non posso desiderare cognata migliore. Grazie anche a te.

Adesso è il momento di andare dietro le quinte. Quando un ragazzo va via di casa, va incontro a nuove avventure, ma se da un lato ci guadagna, dall'altro ci perde. Non è banale lasciare le proprie radici, le certezze, la sua comfort zone e tutto ciò che ti dà casa.

Allontanarsi da casa può far soffrire ma ti apre gli occhi. Quel concetto che dai per scontato all'improvviso non lo è più e ti rendi conto di quanto la famiglia sia importante.

Non mi sarei mai aspettato di avervi tutti qui. È una cosa importantissima per me. Mi fa davvero piacere. Un ringraziamento speciale va ai miei nonni, lo sforzo che stanno facendo oggi i miei nonnini Angela e Carmelo non è paragonabile a nulla. A loro devo tantissimo di quello che sono. Mia nonna mi ha sempre capito con uno sguardo e quando c'è qualcosa che non va, senza sapere nulla mi chiama. Poi c'è mio nonnino, lui mi ha sempre spronato a dare il massimo e non farmi sopraffare dagli altri. Grazie per essere venuti qui nonostante le limitazioni. Loro sono la dimostrazione che l'amore riesce a superare ogni ostacolo.

Devo ringraziare i miei zii, che senza fare troppo rumore hanno assistito a tutta la mia vita. Sono stati sempre presenti ad ogni avvenimento importante della mia vita e tutti loro sempre pronti a consigliarmi, senza aspettarsi nulla in cambio. Da tutti voi ho appreso tanti valori importantissimi. Non riesco davvero a immaginarmi una festività senza di voi. Grazie anche a voi per avermi accompagnato in questo tragitto. Grazie zia Teresa, grazie zia Ivana, grazie zio Gianmarco e grazie zio Mario.

Devo ringraziare anche i miei cuginetti, i quali ho visto tutti nascere e a quanto pare per detto di qualcuno a volte ho pure bullizzato, tipo Simone e credo pure Carlo. Credo che faccia parte del gioco di avere dei cugini più grandi. Però, oh, noi più grandi avevamo sempre la pressione di non dare il cattivo esempio. Se da un lato togli, dall'altro prendi. Grazie Simone, grazie Marta, grazie Carlo, grazie Sara e grazie Isabella.

Ringrazio ovviamente anche tutto il resto della mia famiglia che oggi non è potuto essere presente.

Sono davvero felice di far parte di questa famiglia.

E adesso il pezzo forte, dove tutto è nato. Mio babbo e mia mamma. Avete puntato sempre tutto su noi figli, non saprei quantificare nemmeno i sacrifici che avete fatto per noi e che non ci avete mai fatto pesare per mezzo secondo. Avete avuto sempre pazienza e soprattutto speranza. Grazie per averci insegnato il valore del lavoro, della dedizione e dell'amore incondizionato. Ogni successo che ho ottenuto è frutto del vostro impegno e del vostro sostegno incrollabile. Vi devo tutto ciò che sono e tutto ciò che sarò. Grazie di cuore.

Ora c'è da dire che non so cosa voglia dire avere un solo figlio favoloso e perfetto, ma penso che sia una bella sensazione. Sono arrivato a una conclusione: per essere così favoloso e perfetto, non credo sia solo opera mia, devo ringraziare voi che siete riusciti a forgiare una tale meraviglia. Quindi grazie ancora.

Adesso devo ringraziare l'ultima persona, mio bro Lorenzo. La parte più brutta della partenza per Pisa è stata lasciarti. Avevi appena compiuto 4 anni. Per te sarà una vita, per me sembra ieri. Mi ricordo che mi imitavi su tutto, e quando ti portavo in mezzo alle ragazze sapevi come comportarti. Ogni volta che tornavo ti ritrovavo sempre più alto e grande. Mi dispiace non averti vissuto al 100% e mi sono sempre preoccupato di dare il giusto esempio al mio fratellino. Ogni volta che avrei voluto mollare qualcosa mi sono sempre imposto di non farlo per la paura che potessi prendere da me un cattivo esempio. Quindi in tutti i sensi hai contribuito a darmi la forza e sono qui anche grazie a te. Grazie.